НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичних методів захисту інформації

«До захисту допущено»
В.о. завідувача кафедри
_____ Михайло САВЧУК
«__» _____ 2021 р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності: 113 Прикладна математика
на тему: **«Побудова порогової проксі схеми цифрового підпису з використанням примітивів багатовимірної криптографії»**

Виконав: студент 4 курсу, групи ФІ-74
Чашницька Марина Геннадіївна

Керівник: к.ф.-м.н., ст. викладач кафедри ММЗІ Фесенко А.В.

Консультант: _ _____

Рецензент: к.т.н., доц. кафедри ІБ Стьопочкіна І.В. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ — 2021

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичних методів захисту інформації

Рівень вищої освіти — перший (бакалаврський)
Спеціальність (освітня програма) — 113 Прикладна математика,
ОПП «Математичні методи криптографічного захисту інформації»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Михайло САВЧУК
«__» _____ 2021 р.

ЗАВДАННЯ
на дипломну роботу

Студент: Чашницька Марина Геннадіївна

1. Тема роботи: «*Побудова порогової проксі схеми цифрового підпису з використанням примітивів багатовимірної криптографії*»,

керівник: к.ф.-м.н., ст. викладач кафедри ММЗІ Фесенко А.В.,

затверджені наказом по університету №__ від «__» _____ 2021р.

2. Термін подання студентом роботи: 04 червня 2021р.

3. Вихідні дані до роботи: *статті*

4. Зміст роботи:

1) проведено огляд джерел за схемами підпису багатовимірної криптографії і схемами зі спеціальними властивостями;

2) досліджено особливості побудови схем цифрового підпису зі спеціальними властивостями;

3) побудувано загальну порогову проксі схему цифрового підпису з використанням примітивів багатовимірної криптографії;

4) побудувано порогову проксі схему цифрового підпису на основі схеми підпису LUOV;

5) проаналізовано коректність і захищенність побудованих порогових проксі схем цифрового підпису.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): *Презентація доповіді*

6. Дата видачі завдання: 10 вересня 2020 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2020 р.	Виконано
2	Огляд опублікованих джерел за тематикою дослідження	Вересень-жовтень 2020 р.	Виконано
3	Дослідження цифрових підписів зі спеціальними властивостям у багатовимірній криптографії	Листопад-грудень 2020 р.	Виконано
4	Побудова загальної порогової проксі схеми цифрового підпису з використанням примітивів багатовимірної криптографії	Січень-лютий 2020 р.	Виконано
5	Побудова порогової проксі схеми цифрового підпису на основі схеми підпису LUOV	Березень-квітень 2020 р.	Виконано
6	Аналіз отриманих результатів	Травень 2020 р.	Виконано
7	Оформлення пояснювальної записки до роботи	Травень-червень 2020 р.	Виконано

Студент _____ Чашницька М.Г.

Керівник _____ Фесенко А.В.

РЕФЕРАТ

Кваліфікаційна робота містить: 66 стор., 35 джерел.

Метою роботи є побудова порогової проксі схеми цифрового підпису з використанням примітивів багатовимірної криптографії. Об'єктом дослідження є інформаційні процеси в системах захисту інформації. Предметом дослідження є побудова і аналіз коректності та захищенності порогової проксі схеми цифрового підпису на основі промітивів багатовимірної криптографії.

В роботі досліджено схеми цифрових підписів у багатовимірній криптографії та актуальність використання цифрових підписів зі спеціальними властивостями, зокрема порогового підпису та проксі підпису. Розглянуто методи поєднання схеми цифрового підпису багатовимірних криптосистем з відкритим ключем та схеми цифрового підпису зі спеціальними властивостями з дослідженням коректності і стійкості такої схеми цифрового підпису. Побудовано порогову проксі схему цифрового підпису з використанням примітивів багатовимірної криптографії та порогову проксі схему цифрового підпису на основі схеми підпису LUOV. Виконано аналіз коректності та захищенності побудованих схем цифрового підпису.

БАГАТОВИМІРНА КРИПТОГРАФІЯ, ПРОКСІ ПІДПИС, ПОРОГОВИЙ ПІДПИС

ABSTRACT

Qualification work contains: 66 pages, 35 sources.

The aim of the work is to build a threshold proxy scheme of digital signature using primitive multidimensional cryptography. The object of study is information processes in information security systems. The subject of the study is the construction and analysis of correctness and security of the threshold proxy of the digital signature scheme based on the promotes of multidimensional cryptography.

The schemes of digital signatures in multidimensional cryptography and the relevance of using digital signatures with special properties, in particular threshold signature and proxy signature, are investigated in the work. Methods of combining the digital signature scheme of multivariate public key cryptosystems and the digital signature scheme with special properties with the study of the correctness and stability of such a digital signature scheme are considered. A digital proxy threshold scheme using multivariate cryptography primitives and a digital signature threshold proxy based on the LUOV signature scheme are constructed. An analysis of the correctness and security of the constructed digital signature schemes is performed.

MULTIVARIATE CRYPTOGRAPHY, PROXY SIGNATURE, THRESHOLD SIGNATURE

ЗМІСТ

ВСТУП

Актуальність дослідження. У 1988-му році американський лауреат Нобелівської премії з фізики Річард Фейнман створив концепцію квантового компьютера. Він припустив, що квантові комп'ютери зможуть використовувати постулати квантової механіки, завдяки чтому можна буде швидко масштабувати обсяг обчислень, виконуваний класичними комп'ютерами.

В 1990-их роках почалися дослідження квантових обчислень, Пітер Шор запропонував квантовий алгоритм факторизації цілих чисел і вирішення дискретного логарифмування. Цей алгоритм пізніше назвали на його честь. Завдяки цьому алгоритму можно вирішувати ці задачі суттєво швидше, ніж за домопомогою алгоритмів в класичній моделі обчислень. Це означало, що квантовий компьютер може виконувати злам майже усієї асиметричної криптографії, яка використовується на сьогодні.

Так як існує висока ймовірність того, що у найближчий час буде розроблений повномасштабний квантовий комп'ютер, зростає актуальність досліджень в області посквантової криптографії. Тому потрібні альтернативи тим класичним асиметричним криптосистемам, які складність яких грунтується на важких математичних задачах, що не матимуть ефективного розв'язку у квантовій моделі очислень (так звані постквантові криптосистеми). Багатовимірна криптографія є одним з основних кандидатів для створення постквантових криптосистем. Оскільки скоріш за все постквантові криптосистеми прийдуть на заміну старим, в області цифрових постквантових підписів існує багато ефективних та захищених схем, однак бракує схем підписів зі спеціальними властивостями, такими як сліпі, кільцеві та проксі підписи. У сучасному світі існує багато випадків, коли потрібно використовувати цифрові підписи саме зі спеціальними властивостями, зокрема порогової проксі схеми цифрового підпису з використанням примітивів

багатовимірної криптографії.

Метою дослідження є побудова порогової проксі схеми цифрового підпису з використанням примітивів багатовимірної криптографії. Для досягнення мети необхідно вирішити такі завдання:

1) провести огляд джерел за схемами підпису багатовимірної криптографії і схемами зі спеціальними властивостями;

2) дослідити особливості побудови схем цифрового підпису зі спеціальними властивостями;

3) побудувати загальну порогову проксі схему цифрового підпису з використанням примітивів багатовимірної криптографії;

4) побудувати порогову проксі схему цифрового підпису на основі схеми підпису LUOV;

5) проаналізувати коректність і захищенність побудованих порогових проксі схем цифрового підпису.

Об'єктом дослідження є інформаційні процеси в системах захисту інформації.

Предметом дослідження є побудова і аналіз коректності та захищенності порогової проксі схеми цифрового підпису на основі промітивів багатовимірної криптографії.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: методи лінійної та абстрактної алгебри, теорії імовірностей, теорії складності.

Наукова новизна отриманих результатів полягає у тому що вперше було розроблено загальну порогову проксі схему цифрового підпису з використанням примітивів багатовмірної криптографії та порогову проксі схему цифрового підпису на основі схеми підпису LUOV.

Практичне значення результатів полягає у тому, що побудована порогова проксі схема цифрового підпису на основі схеми підпису LUOV та загальна порогова проксі схема цифрового підпису, яка може бути використана для побудови інших схем порогових проксі підписів з використанням примітивів багатовимірної криптографії.

Апробація результатів та публікації. Результати дослідження, описані в роботі, частково були освітлені в доповіді на XIX Всеукраїнській науково-практичній конференції студентів, аспірантів та молодих учених «Теоретичні та прикладні проблеми фізики, математики й інформатики» (м. Київ, 11-13 травня 2021 р.).

1 ОГЛЯД СХЕМ ЦИФРОВИХ ПІДПИСІВ БАГАТОВИМІРНОЇ КРИПТОГРАФІЇ

Для протидії квантовим атакам потрібно винаходити нові класи криптосистем. Стійкість цих систем не повинна ґрунтуватися на складності задачі факторизації цілих чисел або дискретного логарифмування. Такі криптосистеми мають назву постквантових, тобто такі криптосистеми, які будуть стійкими навіть при застосуванні квантового комп'ютера з великою кількістю кубітів.

У 2006 році була проведена перша конференція на тему постквантової криптографії (PostQuantumCrypto 2006). Ця конференція проводиться щорічно і на ній обговорюються останні досягнення у напрямку постквантової криптографії. Зараз можливо виокремити такі класи постквантових криптосистем, які є стійкими до квантового криптоаналізу [1]:

– Криптографія на основі решіток.

– Багатовимірна криптографія.

– Криптографія на основі геш-функцій.

– Криптографія на основі кодів.

– Криптографія ізогеній суперсингулярних еліптичних кривих.

– Симетрична криптографія.

За даними Національного інституту стандартів і технологій (NIST), багатовимірна криптографія є одним з основних кандидатів в процесі стандартизації, тобто одним із кандидатів для заміни, наприклад, криптосистема RSA [2]. Багатовимірні схеми в цілому дуже швидкі і вимагають лише незначних обчислювальних ресурсів, що робить їх привабливими для використання на недорогих пристроях, таких як смарт-карти та RFID-чіпи [3, 4].

1.1 Багатовимірна криптосистема з відкритим ключем та схема цифрового підпису

Багатовимірна криптосистема з відкритим ключем (МРКС) – це криптосистема з відкритим ключем, де $\mathcal{P}$ – одностороння функція з секретом, задається як набір m багаточленів невеликих степенів d, які складаються з n змінних у скінченному полі $\mathcal{F}$. Зазвичай $d = 2$, тому альтернативною назвою є «Multivariate Quadratic» (MQ). Щоб розшифрувати, автентифікувати або створити цифровий підпис, потрібно для заданого кортежу $z = (z_1, \ldots, z_m)$ знайти розв'язок для $w = (w_1, \ldots, w_n)$ системи

$$\mathcal{P} \begin{cases} p_1(w_1, \ldots, w_n) = z_1 \\ \ldots \\ p_m(w_1, \ldots, w_n) = z_m \end{cases}.$$

Для цифрового підпису, схеми автентифікації "запит-відповідь"та схеми шифрування, $z = (z_1, \ldots, z_m)$ є гешем повідомлення, яке треба підписати, запит та шифротекст відповідно, а $w = (w_1, \ldots, w_n)$ – це підпис цього повідомлення, відповідь та відкритий текст. Кожен може перевірити, чи відповідає дана пара (w, z) системі $\mathcal{P}$, або обчислити $z = \mathcal{P}(w)$ для будь-якого заданого w. Проте знайти хоча б один розв'язок $w = (w_1, \ldots, w_n)$ для системи $\mathcal{P}$ повинно бути складним для більшості $z = (z_1, \ldots, z_m)$ без знання секретного ключа, але легко, знаючи секретний ключ . Тобто, не знаючи секретного ключа, неможливо розшифрувати повідомлення, підробити підпис або пройти автентифікацію.

В загальному випадку задача знаходження коренів системи квадратних рівнянь над скінченним полем є NP-складною для будь-якого скінченного поля. Вона називається $\mathcal{MQ}$-задачею; можливо вона є навіть імовірнісно складною: при $m,n \to \infty$, $\forall \varepsilon > 0$, для будь-якої

імовірнісної машини Тюрінга $\mathcal{A}$ $Pr(\mathcal{P}(x) = y$ розв'язується за допомогою $\mathcal{A}$ за час $poly(m,n)) < \varepsilon$.

Проте важко довести, що багатовимірні схеми є захищеними, бо система $\mathcal{Q}$ повинна бути простою для розв'язання, тому вона не є випадковою, і жодна система $\mathcal{P}$ не отримується шляхом лінійних змін змінних системи $\mathcal{Q}$. Завжди є варіант розв'язати систему рівнянь безпосередньо на основі базисів Грьобнера (XL, F_4, F_5). Для більш-менш «випадкових» рівнянь більшість сучасних методів вимагають експоненціальний час, коли m і n однакового розміру. За методами базисів Грьобнера, декотрі системи є досить невипадковими. Нещодавно до багатовимірних схем (МРКС) були застосовані методи диференціального аналізу і у результаті була зламана схема SFLASH [5, 6].

Використовуючи алгоритм Шора, не можна прискорити розв'язок $\mathcal{MQ}$-задачі, через це, на відміну від багатьох сучасних криптосистем, наприклад, криптосистема RSA, криптосистеми на еліптичних кривих тощо, які ламаються квантовим комп'ютером, багатовимірні криптосистеми з відкритим ключем є кандидатами на постквантову криптографію.

Існують модифікації багатовимірних криптосистем з відкритим ключем,в яких використовуються системи поліномів із змішаними змінними w і z ("неявний МРКС"), або такі, де не усі рівняння системи $\mathcal{P}$ повинні бути дійсними, тільки якийсь визначений відсоток ("ймовірнісний МРКС").

Як правило, багатовимірна схема будується з легко розв'язуваної системи $\mathcal{Q}(x) = y$, головного відображення, яке потім "приховується"двома секретними випадковими лінійними (або афінними) відображеннями $\mathcal{S} : w \mapsto x$ і $\mathcal{T} : y \mapsto z$, щоб отримати нову систему $\mathcal{P}$, зробивиши композицію (зліва і зправа) з $\mathcal{Q}$. Система $\mathcal{P}$ – це відкритий ключ, система $\mathcal{Q}$ і відображення S і T утворюють секретний ключ. $\mathcal{Q}$ задано за допомогою m поліномів $(q_1, \ldots, q_m)$ у $(x_1, \ldots, x_n)$, нова система

$\mathcal{P}$ задається за допомогою m поліномів $(p_1, \ldots, p_m)$ у $(w_1, \ldots, w_n)$ як у: $(p_1, \ldots, p_m) \cdot (w_1, \ldots, w_n) = \mathcal{T}(q_1(\mathcal{S}(w_1, \ldots, w_n)), \ldots, q_m(S(w_1, \ldots, w_n)))$, де $\mathcal{S}$ і $\mathcal{T}$ - таємні випадкові лінійні (або афінні) бієктивні відображення змінних. Нова система $\mathcal{P} := \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$ також квадратична і знайти розв'язок системи $\mathcal{P}$ повинно бути важко без знання $\mathcal{S}$ і $\mathcal{T}$.

Існує кілька сімейств багатовимірних схем, що відповідають декільком виборам для системи $\mathcal{Q}$. Деякі схеми були зламані; для інших застосовуються лише загальні атаки на основі базисів Грьобнера, і параметри вибираються досить великими, щоб зробити злам неможливим.

1.2 Огляд схеми цифрового підпису UOV

У 1997 році Ж. Патарін запропонував схему, яка називається Oil and Vinegar(Олія і оцет) для криптографії на основі відкритих ключів [7]. Ця схема використовує багатовимірні квадратичні поліноміальні рівності. над маленьким скінченним полем як відкритий ключ і схожі поліноми як секретний ключ. В схемі OV складність зламу забезпечувалася структурою багатовимірних квадратичних поліномів $p_i^{'}$. Нехай $o \in \mathbb{N}$ це кількість змінних "олії", а $v \in \mathbb{N}$ це кількість змінних "оцту".

В схемі UOV система $\mathcal{Q}$ задається o квадратичних багаточленів з $n = o + v$ змінними $(w_1, \ldots, w_o)$, $(w_{o+1}, \ldots, w_{o+v})$. Перші змінні $w_1, \ldots, w_m$ називали змінними "олії", а інші змінні $w_{o+1}, \ldots, w_{o+v}$ називали змінними "оцту". Кожен поліном може містити квадратичні члени форми "олія $\times$ оцет" $(w_i w_j)$ з $(1 \leqslant \text{i} \leqslant o \text{ і } o+1 \leqslant \text{i} \leqslant o+v)$ або "оцет $\times$ оцет" ($w_i w_j$ з $o+1 \leqslant i, j \leqslant o+v$), але не повинні містити квадратних термінів форми "олія $\times$ олія"($w_i w_j$ з $1 \leqslant i, j \leqslant o$). Як наслідок, систему $\mathcal{Q}$ легко розв'язати, зафіксувавши довільні значення для оцтових змінних та розв'язавши отриману систему (з o змінними "олії") шляхом вилучення Гауса, однак система $\mathcal{P}$ повинна приховувати різницю між змінними олії та оцту. Є $n = o + v$, більш того, $m = o$ і $o = v$ (або також $n = 2m$) для випадку схем OV. Секретні поліноми $p_i^{'}$ для $1 \leqslant i \leqslant m$ можуть бути

представлені у вигляді:

$$p_i^{'}(x_1^{'},\ldots,x_n^{'}) = \sum_{1\leqslant j\leqslant v, 1\leqslant k\leqslant n} \gamma_{i,j,k} x_j^{'} x_k^{'} \sum_{1\leqslant k\leqslant n} \beta_{i,k} x_k + \alpha_i, \text{ для } 1 \leqslant i \leqslant m,$$

$1 \leqslant j \leqslant v$ і $1 \leqslant k \leqslant n$, $\alpha_i, \beta_{i,k}, \gamma_{i,j,k} \in \mathbb{F}$.

Складність зламу забезпечується афінним відображенням $S \in AGL_n(\mathbb{F})$, яке змішує змінні "олії" і "оцту". Щоб розв'язати таку систему, законному користувачу потрібно усі змінні "оцту" обрати випадковими. Таким чином він отримає випадкову і лінійну рівність зі змінних "олії", які можна розв'язати вилученням Гауса.

Загалом, схема OV(UOV) зроблена як схема підпису. Вона не підходить для шифрування, тому що параметр v вибирається занадто великим для належного рівня захищеності. Щоб підписати повідомлення $M \in \mathbb{F}^m$, треба виконати наступні кроки:

1. Присвоюються випадкові значення $a_1,\ldots,a_s$ для всіх змінних "оцту".

2. Після підстановки випадкових значень система $M = \mathcal{P}^{'}(a)$ стає лінійною. Ця лінійна система розв'язується для m змінних $a_1,\ldots,a_o$ з a, які залишилися, за допомогою вилучення Гауса. Якщо лінійна система сингулярна, повертаємося до першого кроку і пробуємо ще раз для нових випадкових змінних "оцту".

3. Використовується розв'язок a для підпису x: $x = S_{-1}(a)$.

Перевіряємо, що підпис $x \in \mathbb{F}^n$ це розв'язок відкритої системи $\mathcal{P}$. Зловмисник, який хоче підмінити підпис для повідомлення $M = (M_1,\ldots,M_m)$ повинен розв'язати наступну систему:

$$\begin{cases} M_1 = p_1(x_1,\ldots,x_n) \\ \ldots \\ M_m = p_m(x_1,\ldots,x_n) \end{cases}.$$

Загалом, це MQ-задача, яку складно розв'язати. Оскільки оригінальна схема OV була заламана [8], А. Кіпніс в роботі [9] зробив схему UOV. Для цієї схеми потрібно, щоб $v > o$ (чи $n > 2m$).

1.3 Дослідження схеми цифрового підпису Rainbow

Ідея схеми Rainbow [10] полягає у використанні u екземплярів UOV ітеративним способом. Перший екземпляр UOV містить $v_2 - v_1$ поліномів у $v_2 - v_1$ змінних "олії" та v_1 змінних "оцту". Другий екземпляр UOV містить $v_3 - v_2$ поліномів з $v_3 - v_2$ змінними "олії" та v_2 змінними "оцту". Останній UOV екземпляр містить $v_{u+1} - v_u$ поліномів з $v_{u+1} - v_u$ змінними "олії" та v_u змінними "оцту". Позначаючи v_{u+1} через n, отримана система $\mathcal{Q}$, таким чином, містить $m = n - v_1$ поліномів з n змінних, і її легко розв'язати рекурсивним застосуванням принципу UOV: зафіксувати довільні змінні v_1 перших екземплярів UOV, розв'язати цей екземпляр у змінних $v_2 - v_1$, потім розглянути всі $v_1 + (v_2 - v_1) = v_2$ змінні як змінні "оцту" другого екземпляру UOV тощо. Це означає, що законний підписувач повинен вирішувати більше систем, але системи є меншими порівняно з UOV. Для ретельно підібраних параметрів схема Rainbow залишається стійкою. TTS [11], попередник Rainbow, можна вважати агресивною варіацією Rainbow з розрідженими коефіцієнтами.

Схема підпису Rainbow [12] є однією з найбільш перспективних та найкраще вивчених багатовимірних схем підпису. Схему можна описати наступним чином:

Нехай $\mathbb{F} = \mathbb{F}_q$ – скінченне поле з q елементів, $n \in \mathbb{N}$ та $v_1 < v_2 < \ldots < v_l < v_{l+1} = n$ – послідовність цілих чисел. Встановимо $m = n - v_1$, $O_i = v_i + 1, \ldots, v_{i+1}$ і $V_i = 1, \ldots, v_i (i = 1, \ldots, l)$.

Створення ключів: секретний ключ схеми складається з двох бієктивних афінних відображень $\mathcal{S} : \mathbb{F}^m \to \mathbb{F}^m$ і $\mathcal{T} : \mathbb{F}^n \to \mathbb{F}^n$ та квадратного відображення $\mathcal{F}(x) = (f^{(v_1+1)}(x), \ldots, f^{(n)}(x) : \mathbb{F}^n \to \mathbb{F}^m$.

Поліноми $f^{(i)} (i = v_1 + 1, \ldots, n)$ мають вигляд

$$f^{(i)} = \sum_{k,l \in V_j} \alpha_{k,l}^{(i)} \cdot x_k \cdot x_l + \sum_{k \in V_j, l \in O_j} \beta_{k,l}^{(i)} \cdot x_k \cdot x_l + \sum_{k \in V_j \cup O_j} \gamma_k^{(i)} \cdot x_k + \eta^{(i)}$$

з коефіцієнтами, обраними випадковим чином з $\mathbb{F}$. Тут j є єдиним цілим числом, таким що $i \in O_j$. Відкритий ключ – це композиція

відображенння $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T} : \mathbb{F}^n \to \mathbb{F}^m$.

Створення підпису: Щоб сформувати підпис для документа $w \in \mathbb{F}^m$, обчислюється рекурсивно $x = \mathcal{S}^{(-1)}(w) \in \mathbb{F}^m$, $y = \mathcal{F}^{(-1)}(w) \in \mathbb{F}^n$ та $z = \mathcal{T}^{(-1)}(y)$. Це робиться, як показано в Алгоритмі 1.

Алгоритм 1.1. Обернення головного відображення Rainbow

Вхід: Головне відображення $\mathcal{F}$, вектор $x \in \mathbb{F}^m$.

Вихід: Вектор $y \in \mathbb{F}^n$ такий, що $\mathcal{F}(y) = x$.

1: Вибираються випадкові значення для змінних $y_1, \ldots, y_{v_1}$ і підставляються ці значення до багаточленів $f^{(i)}(i = v_1 + 1, \ldots, n)$.

2: **for** $k = 1$ to l **do**

3: Виконується вилучення Гауса на поліномах $f^{(i)}(i \in O_k)$, щоб отримати значення змінних $y_i(i \in O_k)$.

4: Підставляється значення $y_i(i \in O_k)$ у багаточлени $f^{(i)}, i \in v_{k+1} + 1, \ldots, n$.

5: **end for**

Може трапитися так, що одна з лінійних систем на кроці 3 алгоритму 1.1 не має розв'язку. У цьому випадку потрібно обрати інші значення для $y_1, \ldots, y_{v_1}$ і розпочати знову. Підпис документа w дорівнює $z \in \mathbb{F}^n$.

Перевірка підпису: Щоб перевірити справжність підпису $z \in \mathbb{F}^n$, просто обчислюється $w' = \mathcal{P}(z) \in \mathbb{F}^m$. Якщо виконується $w' = w$, підпис приймається, інакше – відхиляється.

1.4 Приклад схеми сліпого підпису на основі Rainbow

В схемі сліпого підпису підписувачі мають власні секретні ключі для підписування і поширюють свої відкриті перевірочні ключі, як у звичайній схемі цифрового підпису [13]. Відкриті перевірочні ключі поширюються через автентифіковані канали, наприклад, методами інфраструктури відкритих ключів. Також є відкритий алгоритм

перевірки, такий що будь-хто, хто має відкритий ключ y підписувача може перевірити чи поданий підпис s є справжнім для повідомлення m згідно з перевірочним ключем y підписувача.

У схемі сліпого підпису підписувачі не можуть дізнатися повідомлення, які вони підписують, а також підписи, які вони отримують. Перевіряльник отримує підпис для повідомлення m від підписувача з перевірочним ключем y, проводить деякі дії з повідомленням m та передає m підписувачеві. Підписувач дає відповідь s отримувачу, таку що отримувач може отримати підпис $s^{'}$ з $y, m, m^{'}, s$, такий що $s^{'}$ справжній для $m^{'}$ згідно з y. Фінальний підпис $s^{'}$ називається "сліпий підпис хоча це не підпис є сліпим, а підписувач.

Сліпий підпис – це підпис, який має властивість, яка дає приватність та захищає від обману чесних одержувачів, забезпечує співпрацю підписувачів та перевіряльник.

Сліпий цифровий підпис на основі Rainbow

У цьому розділі дається детальний опис схеми сліпого підпису на основі Rainbow. Схема складається трьох алгоритмів *KeyGen*, *Sign* і *Verify*, де *Sign* – це інтерактивний протокол між користувачем і підписувачем.

Параметри: Скінченне поле $\mathbb{F}$, цілі числа m, n і r (залежно від захищеного параметра k). r тут визначає, скільки раундів схеми ідентифікації виконувалось під час генерації підпису.

Створення ключів: Підписувач вибирає випадковим чином секретний ключ Rainbow (складається з двох афінних відображень $\mathcal{S} : \mathbb{F}^m \to \mathbb{F}^m$ і $\mathcal{T} : \mathbb{F}^n \to \mathbb{F}^n$ та секретне головне відображення $\mathcal{F} : \mathbb{F}^n \to \mathbb{F}^m$). Він обчислює відкритий ключ $\mathcal{P}$ як $\mathcal{P} = \mathcal{S} \circ \mathcal{F} \circ \mathcal{T} : \mathbb{F}^n \to \mathbb{F}^m$ і використовує CSPRNG для формування системи $\mathcal{R} = CSPRNG(\mathcal{P}) : \mathbb{F}^m \to \mathbb{F}^m$. Відкритим ключем схеми сліпого підпису є пара $(\mathcal{P}, \mathcal{R})$, секретний ключ підписувача складається з $\mathcal{S}, \mathcal{F}$ і $\mathcal{T}$. Однак, оскільки $\mathcal{R}$ можна обчислити з системи $\mathcal{P}$, не потрібно публікувати $\mathcal{R}$

Створення підпису: Щоб отримати підпис для повідомлень із

геш-значенням $\mathcal{H}(d) = w \in \mathbb{F}^m$, користувач вибирає випадково вектор $z^* \in \mathbb{F}^m$. Він обчислює $w^* = \mathcal{R}(z^*) \in \mathbb{F}^m$ і посилає $\tilde{w} = w - w^* \in \mathbb{F}^m$ підписувачеві. Підписувач використовує свій секретний ключ $(\mathcal{S}, \mathcal{F}, \mathcal{T})$ для обчислення підпису $z \in \mathbb{F}^n$ такого, що $\mathcal{P}(z) = \tilde{w}$, і відправляє z назад користувачеві, який, таким чином, отримує розв'язок (z, z^*) системи $\bar{\mathcal{P}}(x) = \mathcal{P}(x_1) + \mathcal{R}(x_2) = w$. Щоб довести ці знання перевіряльнику з нульовим розголошенням, користувач генерує підпис MQDSS для повідомлення w. Як загальнодоступний параметр схеми він тут використовує систему $\bar{\mathcal{P}}(x) = \mathcal{P}(x_1) + \mathcal{R}(x_2)$, яка є системою m квадратних рівнянь у $n + m$ змінних. Крім того, $\mathcal{G}(x, y)$ є полярною формою системи $\mathcal{P}$, тобто $\mathcal{G}(x, y) = \bar{\mathcal{P}}(x + y) - \bar{\mathcal{P}}(x) - \bar{\mathcal{P}}(y) + \bar{\mathcal{P}}(0)$. Зокрема, користувач виконує наступні кроки.

1. Використовує загальновідому геш-функцію H для обчислення $\mathcal{C} = \mathcal{H}(\mathcal{P}||w)$ і $\mathcal{D} = \mathcal{H}(\mathcal{C}||w)$.

2. Вибирає випадкові значення для $r_{01}, \ldots, r_{0r}, t_{01}, \ldots, t_{0r} \in \mathbb{F}^{m+n}, e_{01}, \ldots, e_{0r} \in \mathbb{F}^m$, встановлює $r_{1i} = (z||z^*) - r_{0i} (i = 1, \ldots, r)$ та обчислює зобов'язання $c_{0i} = Com(r_{0i}, t_{0i}, e_{0i})$ та $c_{1i} = Com(r_{1i}, \mathcal{G}(t_{0i}, r_{1i}) - e_{0i}) (i = 1, \ldots, r)$. Встановлює $COM = (c_{01}, c_{11}, c_{02}, c_{12}, \ldots, c_{0r}, c_{1r})$.

3. Виводить значення $\alpha_1, \ldots, \alpha_r \in \mathbb{F}$ з $(\mathcal{D}, COM)$.

4. Обчислює $t_{1i} = \alpha_i \cdot r_{0i} - t_{0i} \in \mathbb{F}^{m+n}$ та $e_{1i} = \alpha_i \cdot \bar{P}^{(r_{0i})} - e_{0i} (i = 1, \ldots, r)$. Встановлює $Rsp_1 = (t_{11}, e_{11}, \ldots, t_{1r}, e_{1r})$.

5. Виводить значення $(ch_1, \ldots, ch_r)$ з $(\mathcal{D}, COM, Rsp_1)$.

6. Встановлює $Rsp_2 = (r_{ch_1 1}, \ldots, r_{ch_r r}, r)$.

7. Сліпий підпис σ для повідомлення $w \in \mathbb{F}^m$ визначається як $\sigma = (\mathcal{C}, COM, Rsp_1, Rsp_2)$.

Довжина сліпого підпису σ визначається як $|\sigma| = 1 \cdot$ |геш-значення| $+2r \cdot$ |зобов'язання| $+ r \cdot (2n + 3m)\mathbb{F}$-елементи.

Перевірка підпису: Щоб перевірити справжність сліпого підпису σ для повідомлень із геш-значенням $w \in \mathbb{F}^m$, перевіряльник аналізує компоненти σ та обчислює $\mathcal{D} = \mathcal{H}(\mathcal{C}||w)$. Він виводить значення $\alpha_i \in \mathbb{F}$ з

$(\mathcal{D}, COM)$ та ch_i з $(\mathcal{D}, COM, Rsp_1)(i = 1, \ldots, r)$. Нарешті, він аналізує COM на $(c_{01}, c_{11}, c_{02}, c_{12}, \ldots, c_{0r}, c_{1r}), Rsp_1$ в $t1, e1, \ldots, t_r, e_r$ і Rsp_2 у $r_1, \ldots, r_r$ і перевіряє, чи для всіх $i = 1, \ldots, r, r_i$ – правильна реакція на ch_i щодо COM, t_i та e_i, тобто

$c_{0i} \stackrel{?}{=} Com(r_i, \alpha_i \cdot r_i - t_i,\ \alpha_i \cdot \mathcal{P}(r_i) - e_i)$ (для $ch_i = 0$)

$c_{1i} \stackrel{?}{=} Com(r_i, \alpha_i \cdot (w - \mathcal{P}(r_i)) - \mathcal{G}(t_i, r_i) - e_i)$ (для $ch_i = 1$).

Якщо всі ці тести виконані, сліпий підпис σ приймається, інакше – відхиляється.

Оскільки результуючий сліпий підпис залежить від вибірковості випадковості для генерування доказу нульового знання, може бути багато підписів, пов'язаних з одним кортежем (z, z^*). Щоб запобігти зловмисному користувачеві повторне використання тієї самої зображення до $\mathcal{P}(\bar{x_1}) + \mathcal{R}(\bar{x_2})$, два підписи до повідомлень d_1, d_2 вважаються істотно різними, коли $w_1 = \mathcal{H}(d_1) \neq w_2 = \mathcal{H}(d_2)$. Іншими словами, доказ нульового знання враховується для обґрунтованості, але не для чіткості.

Ця схема є дуже ефективною і виробляє значно коротші сліпі підписи, ніж схема, заснована на решітці Рюкерта [14]. Це стосується також Rainbow та MQDSS, їх властивості використовуються лише абстрактно, і цілком можна уявити, що інша комбінація схеми підпису MQ на основі однонаправленої функції з неінтерактивним підтвердженням знання розв'язку системи MQ дасть той самий результат [15].

Одним з основних випадків використання сліпих підписів є анонімна ідентифікація. У цьому сценарії можна обґрунтовано відмовитися від модифікованої схеми підпису, а замість цього безпосередньо використовувати базову інтерактивну схему ідентифікації, дезактифікуючи неінтерактивність для менших обчислень і пропускної здатності. Подібним чином, для інших випадків, таких як доступ до анонімних баз даних, потрібні анонімні облікові дані. Цю схему можна адаптувати відповідно до цього сценарію, просто вказавши, що всі користувачі отримують сліпий підпис на одному і тому ж

загальнодоступному параметрі.

Стійкість сліпого цифрового підпису

Далі наводиться доведення захищеності схеми сліпого цифрового підпису, припускаючи абстрактно, що Rainbow захищена [16]. Для цього треба показати стійкість отриманої схеми.

Теорема 1.1. *Припустимо, що розподіл $\mathcal{R}(x)$ для рівномірного $x \in \mathbb{F}_q^m$ обчислювально не відрізняється від рівномірного, і припустимо, що використовується ідеально приховуюча схема зобов'язань. Тоді багатовимірна схема сліпого підпису забезпечує стійкість проти будь-якого обчислювально обмеженого противника.*

Доведення. Суперник повинен пов'язати $\tilde{w}$ від однієї взаємодії до пари (d, σ) від іншої взаємодії. Через ідеальну властивість нульового розголошення ідеально прихованої схеми зобов'язань, σ не містить інформації про розв'язок (z, z^*), а отже, інформації про $\mathcal{R}(z^*)$ або $\mathcal{P}(z)$. Тому завдання супротивника еквівалентно прив'язуванню $\tilde{w}$ до d, оскільки знання σ не дає йому переваг. Однак z^* вибирається рівномірно навмання і так $\mathcal{R}(z^*)$ обчислювально не відрізняється від рівномірного. Як результат, засліплене повідомлення $\tilde{w} = w\mathcal{R}(z^*)$ обчислювально не відрізнити від рівномірного, і жоден супротивник за поліноміальний час не може обчислити будь-який предикат w з $\tilde{w}$ з більш ніж незначною ймовірністю успіху. Сюди входить предикат $\mathcal{H}(d) \stackrel{?}{=} w$ або будь-який подібний предикат, який би дозволив супротивнику пов'язати $\tilde{w}$ з d. □

Коректність сліпого цифрового підпису

Теорема 1.2. *Сліпі підписи, сформовані чесними учасниками протоколів цієї багатовимірної схеми сліпого підпису, будуть прийняті з імовірністю 1.*

Доведення. Доведення складається з двох частин. Спочатку показується, що в кінці інтерактивного процесу користувач отримує розв'язок (z, z^*) системи $\mathcal{P}(x_1) + \mathcal{R}(x_2) = w$. Це можна побачити наступним чином. У процесі інтерактивного протоколу (чесний)

користувач вибирає випадковим чином вектор z^*, обчислює $w^* = \mathcal{R}(z^*)$ та $\tilde{w} = ww^*$ і відправляє $\tilde{w}$ підписувачеві. (Чесний) користувач підписує свій секретний ключ для обчислення вектора z такого, що $\mathcal{P}(z) = \tilde{w}w$. Разом отримуємо $\mathcal{P}(z) + \mathcal{R}(z^*) = \tilde{w} + w^* = ww^* + w^* = w$, що означає, що (z, z^*) справді є розв'язком публічної системи $\bar{\mathcal{P}} = \mathcal{P}(x_1) + \mathcal{R}(x_2)$. На другому кроці використовується доведення правильності MQDSS [17], щоб показати, що підпис MQDSS, створений чесним підписувачем, що знає розв'язок загальнодоступної системи $\bar{\mathcal{P}}$, чесним перевіряльником приймається з імовірністю 1. □

Висновки до розділу 1

У розділі розглянуто примітиви багатовимірної криптографії та головні задачі, на яких грунтується стійкість багатовимірних криптосистем з відкритим ключем. Визначено особливості основні схеми цифрового підпису багатовимірної криптографії, які полягають у використанні інтерактивних протоколів та відображень зі спеціальними властивостями. Продемонстровано поєднання схеми цифрового підпису багатовимірної криптографії та схеми цифрового підпису зі спеціальними властивостями з дослідженням коректності і стійкості такої схеми цифрового підпису.

2 ОГЛЯД СХЕМ ЦИФРОВИХ ПІДПИСІВ ЗІ СПЕЦІАЛЬНИМИ ВЛАСТИВОСТЯМИ

У цьому розділі розглядаються різні схеми цифрових підписів зі спеціальними властивостями. Наведено огляд адаптації цих схем у багатовимірній криптографії.

2.1 Визначення проксі підпису та приклади схем проксі підписів

Спробуємо уявити цілком звичайну ситуацію. Одному з директорів необхідно виїхати у відрядження, без можливості взяти з собою комп'ютер, за допомогою якого він міг би в повній мірі отримати доступ до корпоративних ресурсів. Але робота компанії не припиняється на час його відрядження, на корпоративну пошту приходить багато листів, потрібно узгоджувати багато речей дистанційно. З цим йому може допомогти заступник, але враховуючи важливість даних, які потрібно узгодити, заступнику потрібно відповідати на усі запити від імені директора, для цього усі відповіді повинні бути підписаними цифровим підписом директора.

Саме для наведеної ситуації існують проксі підписи. До того ж, треба звернути увагу на те, що при наявності тільки однієї пари відкритого й секретного ключа не можна здійснювати їхню передачу іншій особі. Враховуючи це, проксі підпис дозволяє делегувати свою можливість підписувати своєму заступнику – довіреній особі; він вирішує проблему з передачею відкритих і секретних ключів. Це і називається задачею делегування прав, або проксі задачею. Проведено багато роботи для реалізації механізму автентифікації на базі проксі підпису.

Якщо авторизований користувач, наприклад, заступник, може

показати, що він є правильною особою, яку треба уповноважити, то усі дії цього уповноваженого користувача будуть вважатися діями того, хто делегував йому ці повноваження. Цифровий підпис є методом, який, у свою чергу, допомагає надати інформацію про людину, яка підписала документ, а також довести, що цей документ не був зміненим або підробленим, до того ж вказуючи справжній час підписання документу.

Існує три різні типи делегування у проксі підписах: повне делегування, часткове делегування та делегування з гарантією [18].

Повне делегування

У цьому типі делегування довірена особа отримує той самий секретний ключ, яким володіє оригінальний підписувач. Відповідно, якщо довірена особа буде підписувати документ, який оригінальний підписувач, не став би приймати, ніхто не зможе виявити її злочинних дій.

Часткове делегування

У випадку часткового делегування з оригінального секретного ключа S, утворюється новий секретний ключ S', і, відповідно, змінюється рівняння перевірки. Довірена особа отримує новий секретний ключ S' безпечним шляхом. Після цього утворений підпис перевіряється вже новим рівнянням перевірки, а не оригінальним. З цього випливає, що такий підпис можна відрізнити від підпису, який створив оригінальний підписувач, та визначити хто саме підписував документ у випадку, якщо був підписаний документ з неприйнятним змістом. Слід зазначити, що за таких умов проксі підпис для кожної довіреної особи буде різним. Для перевірки у цьому типі делегування потрібен тільки відкритий ключ оригінального підписувача.

Делегування з гарантією

У цьому типі делегування використовується гарантія, яка засвідчує, що B – саме та особа, якій A делегував свої повноваження. Делегування з гарантією виконується за допомогою послідовного підписування схемою з відкритим ключем. Для такого підходу існує два варіанти схем підпису.

У першому випадку повідомлення та оригінальний підпис

використовується для утворення відкритого ключа B. Або гарантія складається тільки з повідомлення, яке оголошує, що є довіреною особою. Отримуючи гарантію, B підписує документ своїм секретним ключем Y звичайною схемою підпису: правильний проксі підпис складається зі створеного підпису разом з гарантією. У цьому випадку новий створений підпис не має відношення до оригінального.

У другому випадку гарантія складається з повідомлення і оригінального підпису для наново створеного відкритого ключа. Секретний ключ, який є сумісним з новим відкритим ключем, передається B безпечним шляхом. Отримавши гарантію, B підписує документ отриманим секретним ключем, використовуючи відповідну схему підпису. Створений проксі підпис такий самий, як у першому випадку, а для його перевірки потрібен тільки відкритий ключ оригінального підписувача.

Підсумовуючи все вище зазначене, найбезпечнішими методами делегування є часткове делегування і делегування з гарантією. Довірена особа ніколи не зможе створити оригінальний підпис у випадку з частковим делегуванням, а делегування з гарантією використовує звичайні схеми підпису без модифікацій. Також, якщо скомбінувати часткове делегування та делегування з гарантією, буде отримано часткове делегування з гарантією.

Модифікації схем проксі підпису

У 1996 М. Якобсон [19] запропонував схему підпису з назначеним перевіряльником (DVS). Цей тип підпису забезпечує автентифікацію повідомлень без відмови. У такій схемі підпису лише назначений перевіряльник може перевірити правильність підпису і він не може переконати у цьому когось іншого, оскільки сам здатен створити підпис, який не може буде розрізнити з тим, правильність якого він перевіряє. До того ж, у 2003 С. Саідніа [20] додав концепт сили до схеми DVS, зробивши з неї сильну схему підпису з назначеним перевіряльник (SDVS). Під цим

мається на увазі, що назначений перевіряльник може підтвердити правильність підпису тільки протягом фази перевірки. SDVS схема Саідніа ґрунтується на складності задачі дискретного логарифмування. Назначений перевіряльник використовує свій секретний ключ під час перевірки. А. Шамір [21] вперше запропонував ідею криптографії з відкритим ключем на основі ID. Криптографія на основі ID дозволяє деяку відкриту інформацію, таку як ім'я, адреса тощо, використовувати як відкритий ключ. У 2005 році М. Хуан [22] запропонував концепт DVS на основі ідентифікаторів. М. Сандер [23] у свою чергу запропонував сильну схему проксі підпису на основі ID (ID-SDVPS), у якій проксі підпис може бути перевірений тільки перевіряльником назначеними оригінальним підписувачем. М. Сандер [24] представив ще одну модифікацію, у якій проксі підпис може бути перевірений тільки двома перевіряльниками назначеними оригінальним підписувачем. Ця схема отримала назву сильної схеми проксі підпису на основі ID з двома назначеними перевіряльниками (ID-based SBDVPS). У своїй роботі він представив 9 схем і зробив порівняльний аналіз їх ефективності.

Огляд ID-SDVPS схеми

У цій схемі використовується центр створення ключів (ЦСК), який обчислює відкритий ключ користувача.

G_1 – група порядку великого простого числа q,

G_2 – мультиплікативна підгрупа скінченного поля $\mathbb{F}$ того самого порядку q,

P – генератор групи G_1,

e: $G_1 \times G_1 \to G_2$ – білінійне відображення.

– **Вхід**: ЦСК обирає генератор $P \in G_1$, випадкове число $s \in {Z_q}^*$ порядку q і обчислює $P_{pub} = sP$.

ЦСК також обирає дві криптографічні геш-функції H_1 та H_2.

$$H_1 : \{0,1\}^* \to G_1,$$
$$H_2 : G_1 \to {Z_q}^*,$$
$$e : G_1 \times G_1 \to G_2.$$

Параметри системи (G_1, G_2, P, P_{pub}, H_1, H_2, e) відкриті, а параметр s зберігає ЦСК.

– **Створення ключа**: Користувач U з ідентифікатором ID_U створює $Q_{ID_U} = H_1(ID_U)$ як свій відкритий ключ і $S_{ID_U} = sQ_{ID_U}$ як свій серкретний ключ.

– **Створення проксі-ключа**: Користувач A обирає два випадкових числа $t \in {Z_q}^*$ і $k \in Z_q$. Обчислює

$$r = e(P, Q_{ID_B})^k,$$
$$U = H_1[rH_1(m, w)],$$
$$V = t^{-1}kP - US_{ID_A}.$$

Він відправляє $\sigma = (m, w, t, r, U, V)$ користувачу B. Отримавши σ, користувач B обчислює

$$U = H_2[rH_1(m,w)]$$

і підтверджує гарантію, якщо

$$[e(V, Q_{ID_B})e(Q_{ID_A}, S_{ID_B})^U]^t = r.$$

Користувач B обчислює секретний проксі-ключ

$$S_{ID_P} = V + S_{ID_B}.$$

– **Створення проксі-підпису**: Щоб створити правильний проксі-підпис для повідомлення m користувач B обирає два випадкових числа t_1

$\in Z_q^*$ і $x \in Z_q$. Обчислює

$$U_1 = t_1^{-1} Q_{ID_C},$$
$$U_2 = xQ_{ID_B},$$
$$h = H_2(m, w, U_1),$$
$$V_1 = t_1(x + h)S_{ID_P},$$
$$V_2 = (x + h)V.$$

Він відправляє $\sigma_1 = (m, w, U_1, U_2, h, V_1, V_2)$ назначеному перевіряльнику C.

– **Перевірка проксі-підпису**: Отримавши σ_1, назначений перевіряльник C виконує наступні кроки:

1) Перевіряє чи повідомлення m відповідає гарантії w. Якщо ні, зупиняється, інакше – продовжує.

2) Перевіряє чи користувачі A та B зазначені як оригінальний підписувач і довірена особа в гарантії w, відповідно.

3) Обчислює

$$h = H_2(m, w, U_1)$$

і підтверджує підпис, якщо

$$e(U_1, V_1) = e(Q_{ID_C}, V_2)e(S_{ID_C}, U_2 + hQ_{ID_B}).$$

Також Сандер представив у 2005 році схему сліпого проксі підпису. Ця схема базується на схемі сліпого підпису Шнора і проксі підпису М. Мамбо.

Схема сліпого проксі підпису

Нижче наведена схема сліпого проксі підпису [25].

Вхід:

p – велике просте число;

q – велике просте число і дільник числа $(p - 1)$;

g – елемент $\mathbb{Z}_p^*$, порядку q;

x_A – секретний ключ оригінального підписувача A;

y_A – відкритий ключ оригінального підписувача A, де $y_A = g^{x_A} mod p$;

$h(\cdot)$ – стійка одностороння геш-функція.

Етап проксі:

1. (Створення проксі) Оригінальний підписувач A випадковим чином обирає $k \in \mathbb{Z}_p^*$, $k! = 1$ і обчислює $r = g^k mod p$, $s = x_A + k \cdot r mod q$ та $y_p = g^s mod p$.

2. (Доставка проксі) Оригінальний підписувач відправляє (s, r) проксі підписувачу B безпечним шляхом і робить y_p відкритим.

3. (Перевірка проксі) Після отримання секретного ключа (s, r), проксі підписувач B перевіряє справжність секретного ключа наступною конгруенцією: $y_p = g^s = y_A \cdot r^r mod p$. Якщо (s, r) задовольняють умови конгруенції, проксі приймається, інакше – відхиляється.

Етап підписування:

1. Користувач B обирає випадкове число $K \in \mathbb{Z}_p^*$, $K! = 1$, обчислює $R = g^K mod p$ і відправляє його користувачу C.

2. Користувач C випадковим чином обирає $\alpha, \beta \in \mathbb{Z}_p^*$ і обчислює $r^{'} = R \cdot g^{-\alpha} \cdot y_p^{-\beta} mod p$. Якщо $r^{'} = 0$, він обирає інший набір $\alpha, \beta \in \mathbb{Z}_p^*$, інакше – обчислює $e^{'} = h(r^{'}, m) mod q$, $e = e^{'} + \beta mod q$ і користувач C відправляє e користувачу B.

3. Після отримання e, користувач B обчислює $s^{'} = K - s \cdot e mod q$ і відправляє його користувачу C.

4. Користувач C обчислює $S_p = s^{'} + \alpha mod q$. Набір (m, S_p, e') – це сліпий проксі підпис.

Етап перевірки:

Перевіряльник або отримувач сліпого прокс-підпису обчислює $e^{''} = (h(g^{S_p} \cdot y_p^{e^{'}} mod p) \oplus m) mod p$. Тут $e^{''} = e^{'}$, тільки якщо підпис (m, S_p, e') правильний.

2.2 Визначення порогового підпису та приклади схем порогових підписів

Пороговий підпис – це цифровий підпис, де підписувачі можуть утворити групу таким чином, що тільки конкретна підгрупа цієї групи може створити підпис від імені групи. [26] Набір підгруп, які мають можливість створити підпис, називається "структурою доступу"порогової схеми. Більш конкретно, порогова схема (t, n) – схема цифрового підпису, де будь-які t або більше підписувачів групи з n учасників можуть створити підпис від імені групи. У загальному випадку пороговий підпис не дає дізнатися, які учасники групи приймали участь у його створенні. [27, 28]

Мультипідпис – це пороговий підпис з додатковою властивістю, яка полягає у тому, що можна дізнатися особи тих, хто приймав участь у створенні підпису. У таких підписах, особи, які створюють підпис взагалі не є анонімними.

Мета порогової схеми підпису полягає у тому, що забезпечується подвійний контроль над можливістю створити підпис ($t > 1$), або виключити можливість відмови ($n > 1$), або обидві ситуації одночасно. Група може вибрати одного учасника для керівництва цією групою (Арбітра), або група може вирішити роздати керівництво усім її учасникам таким чином, що кожен учасник групи приймає участь у реалізації усіх операцій для створення підпису.

В схемі порогового підпису кожен підписувач групи має свою власну пару ключів. Якщо учасники створюють свої пари ключів без необхідності узгоджувати загальні параметри, така порогова схема називається *розподіленою*. [29]

Будь-яка підгрупа з $\geqslant t$ з n учасників групи G можуть створити підпис. Для цього кожен учасник цієї підгрупи передає частковий підпис назначеній особі, а ця особа отримує пороговий підпис з часткових

підписів. Кожен, хто має доступ до ключа відкритої групи G може перевірити порогової підпис. Назначена особа може бути як реальним суб'єктом так і віртуальною організацією, операції якої обчислюються розподіленим чином серед усіх членів групи. [30]

Схема порогового підпису є надійною, якщо назначений перевіряючий може перевірити справжність кожного з часткових підписів перед тим як використати їх для створення порогового підпису. [31]

Схема порогового проксі підпису

Так як проксі підписи не дають контролювати, які саме документи будуть ними підписуватися, був запропонований варіант використання порогового алгоритму розподілення інформації [32]. Проксі підпис може бути створений тільки, якщо певна кількість учасників буде згодна його поставити.

Для цієї схеми був використаний алгоритм Педерсона, який дозволяє вибрати пару ключів (x, y) групі з n учасників, ця група публікує відкритий ключ для встановлення секретного ключа. Група з t учасників може встановити секретний ключ$(t \leqslant n)$.

Оригінальний підписувач: Оригінальний підписувач A генерує випадкове число $r \in \mathbb{Z}_q^*$, обчислює $R = g^r mod p$ і обчислює проксі $s_A = x_A + rRmodq$. Потім генерує багаточлен $f_A(x) = s_A + a_1x + a_2x^2 + \cdots + a_{k-1}x^{k-1} + a_nx^n$; $f_A(0) = s_A$. Кожному учаснику групи підписувачів передається значення відповідно до його номеру $f_A(i)$ по таємному каналу зв'язку. Також Арбітру передається значення старшого коефіцієнта $f_A(x)$.

Група підписувачів: Кожен учасник групи p_i має пару ключів (x_i, y_i) і генерує багаточлен $g_i(x) = x_iY + b_{i1}x + b_{i2}x^2 + \cdots + b_{ik-1}x^{k-1}$, такий що $g_i(0) = Yx_i$. Обчислюється значення багаточлену $g_i(x)$ у точці $j : g_i(j)$. Нехай $G(x) = f_A(x) + \sum_{i=1}^{n} g_i(x)$ – це сума багаточлена оригінального підписувача і багаточлена кожного учасника групи. Кожен і-ий учасник знає таке значення багаточлена $G(x)$ у точці i:

$G(i) = f_A(i) + \sum_{l=1}^{n} g_l(i)$. Секретний ключ це $G(0) = s_A + Y \sum_{l=1}^{n} x_l$.

Підписування: Для підписування використовується алгоритм Шнора формування електронного цифрового підпису. Кожен учасник використовує у якості секретного ключа $G(i)$, обирає випадкове число k_i і обчислює $s_i = k_i \prod_{j=1(j\neq i)}^{k} \frac{i-j}{o-j} + G(i)H(M||Y)$. s_i – частина підпису і-ого учасника. Такою операцією кожен учасник групи множить вільний член $G(x)$ на $H(M||Y)$ і додає k_i, коефіцієнт при старшому члені збільшився на $k_i \prod_{j=1(j\neq i)}^{k} \frac{1}{-j}$. Використовуючи алгоритм Педерсона, отримуємо загальне значення $K = k_1 + k_2 + \cdots + k_t$.

Арбітр: Арбітр отримує від кожного учасника значення s_i. Понижує степінь багаточлена $G(x)$. Була степінь n, а стане $k-1$: $s_i - (a_n + k)i^n$. Отриманий багаточлен позначаємо $G^{'}(x)$. $G^{'}(0) = G(0)H(M||Y) + K$. Отримане значення $G^{'}(0)$ повертається учасникам групи як підпис.

2.3 Дослідження загальної схеми проксі підпису у багатовимірній криптосистемі з відкритим ключем

Дж. Чен представив у своїй роботі [33] постквантову схему проксі підпису, яка грунтується на багатовимірних криптосистемах з відкритими ключами. Він вирішив зосередити свою увагу на підписах зі спеціальними властивостями і дослідив найперспективніші на даний момент схеми цифрових підписів у багатовимірних криптосистемах з відкритим ключем: UOV, Rainbow та MQDSS і створив на їх основі три варіанти схем проксі підпису. Нижче наведено сам алгоритм підпису, який використовується у цих схемах.

Алгоритм 2.1. $Sign(M, sk)$

Вхід: M – повідомлення, $sk = (L_1, F, L_2)$ – секретний ключ

Вихід: σ – підпис повідомлення M

Початок

1: Підписувач обчислює $M_1 = L_1^{-1}(M)$.

2: Підписувач обчислює $M_2 = F^{-1}(M_1)$.

3: Підписувач обчислює $\sigma = L_2^{-1}(M_2)$.

4: Повертається підпис σ.

Кінець

Для цієї схеми алгоритм $Setup(1^\lambda)$ приймає на вхід 1^λ і виводить параметри системи $params$, які головним чином складаються з (q,n,m) і усіх арифметичних операцій над скінченним полем.

Алгоритм створення ключів $KeyGen(params)$ приймає на вхід $params$ і виводить $pk = P$ і $sk = (L_1, F, L_2)$.

Алгоритм перевірки $Verify(\sigma, M, P)$ повертає 1, якщо $P(\sigma) = M$, інакше – повертає 0.

Чен використовував саме варіант проксі підпису з делегуванням з гарантією для адаптації у багатовимірних криптосистемах з відкритим ключем. Оскільки на даний момент існують різні цифрові підписи зі спеціальними властивостями, такі як груповий підпис, сліпий підпис тощо.

На даний момент у багатовимірній криптографії створено три схеми проксі підписів на основі Rainbow, UOV та MQDSS. Нижче наведена загальна схема проксі підпису у багатовимірній криптосистемі з відкритим ключем.

– **Вхід**: Обираємо два додатних числа n та m, $\mathbb{F}_q$ – скінченне поле, і усі операції надалі відбуваються над цим полем. $\mathcal{H} : \{0,1\}^* \to \mathbb{F}_q^n$ – криптографічна геш-функція.

– **Створення ключа**: Цей алгоритм створює відкритий і секретний ключ користувача. Створення ключа відбувається як у схемі підпису багатовимірної криптосистеми з відкритим ключем. Після цього алгоритму секретний ключ користувача A – це $sk_A = (L_{1A}, L_{2A}, F_A)$ і

відкритий ключ $pk_A = P_A$, де $P_A = L_{1A} \circ F_A \circ L_{2A}$. Таким же чином секретним ключем користувача B є $sk_B = (L_{1B}, L_{2B}, F_B)$, а відкритим ключем – $pk_B = P_B$.

– **Делегування**: Користувач A, маючи гарантію $w = (pk_A, pk_B, t)$, виконує наступні кроки, щоб делегувати права підписування користувачу B:

1) Випадковим чином обираємо два афінних відображення $L_1^{'}$ і $L_2^{'}$, які можна обернути і які мають форму L_{1A} і L_{2A}, відповідно. Потім обчислюємо

$$L_1 = L_1^{'} \circ L_{1A}^{-1},$$
$$L_2 = L_{2A}^{-1} \circ L_2^{'},$$
$$P_A^{'} = L_1 \circ P_A \circ L_2.$$

2) Обчислюємо

$$\theta = Sign(\mathcal{H}(w||P_A^{'}), sk_A).$$

3) Відправляємо $(L_1, L_2, P_A^{'})$ і гарантію (w, θ) користувачу B по автентифікованому каналу.

– **Створення проксі ключа**: Маючи $(L_1, L_2, P_A^{'}, w, \theta)$, користувач B виконує наступні кроки:

1) Перевіряємо правильність $P_A^{'}$ і θ. Якщо вони правильні, то переходимо до наступного кроку, інакше – виводимо 0.

2) Випадковим чином обираємо два афінних відображення $L_1^{''}$ і $L_2^{''}$, відповідно, і обчислюємо

$$L_{1p} = L_1 \circ L_{1B}^{-1} \circ L_1^{''-1},$$
$$L_{2p} = L_2 \circ L_{2B}^{-1} \circ L_2^{''-1},$$
$$F_p = L_1^{''} \circ L_{1B} \circ P_A \circ L_{2B} \circ L_2^{''}.$$

3) Обчислюємо підпис σ_{prx} для (w, θ, F_p).

$$\sigma_{prx} = Sign(\mathcal{H}(w||\theta||P_A^{'}), sk_B).$$

4) Отримуємо проксі ключ користувача B
$sk_p = (L_{1p}, L_{2p}, F_p)$, який використовується для створення підписів від імені користувача A, і відкритий ключ $pk_p = P_A^{'}$.

– **Створення проксі підпису**: Користувач B застосовує L_{1p}, L_{2p} і центральне відображенння F_p для базового алгоритму підпису багатовимірної криптосистеми з відкритим ключем, описаному в Алгоритмі 2.1, щоб створити підпис

$$\sigma = Sign(\mathcal{H}(M), sk_p)$$

для повідомлення M і отримує проксі підпис $(\sigma, (w, \theta, P_A^{'}, \sigma_{prx}))$.

– **Перевірка підпису**: Отримуючи $(M, \sigma, (w, \theta, P_A^{'}, \sigma_{prx}))$, кожен може перевірити правильність проксі підпису виконуючи наступні кроки:

1) Перевіряємо правильність θ для гарантії w, використовуючи pk_A.

$$Verify((w, P_A^{'}), \theta, pk_A) \stackrel{?}{=} true.$$

Якщо вони правильні, то переходимо до наступного кроку, інакше – виводимо 0.

2) Перевіряємо правильність σ_{prx} для $(w, \theta, P_A^{'})$, використовуючи pk_B.

$$Verify((w, \theta, P_A^{'}), \sigma_{prx}, pk_B) \stackrel{?}{=} true.$$

Якщо вони правильні, то переходимо до наступного кроку, інакше – виводимо 0.

3) Перевіряємо правильність σ на повідомленні M,

використовуючи $P_A^{'}$.

$$Verify(M, \sigma, P_A^{'}) \stackrel{?}{=} true.$$

Якщо вони правильні, то виводимо 1, інакше – виводимо 0.

Перевіряючий приймає проксі підпис тоді і тільки тоді, коли три умови перевірки проксі підпису на правильність виконуються одночасно. Інакше перевіряючий відхиляє підпис.

Схема проксі підпису на основі UOV

Задання початкових параметрів: Нехай $n, m > 0$, $\mathbb{F}$ – скінченне поле і усі арифметичні операції виконуються на цим полем. $H : \{0,1\}^* \to \mathbb{F}^n$ – криптографічна геш-функція.

Створення ключів: Цей алгоритм створює відкритий і секретний ключі користувача. Створення ключів таке саме як в схемі підпису багатовимірної криптосистеми з відкритим ключем. Після цього алгоритму задається секретний ключ користувача A як $sk_A = (F_A, T_A)$ і відкритий ключ – $pk_A = P_A$, $P_A = F_A \circ T_A$. Таким самим чином секретний ключ користувача B це $sk_B = (F_B, T_B)$, відкритий ключ – $pk_B = P_B$.

Делегування: A випадковим чином обирає бієктивне афінне відображення T, потім обчислює $T_A^{'} = T \circ T_A$, $F_A^{'} = F_A \circ T$ і $P_A^{'} = F_A^{'} \circ T_A^{'}$. Афінна T повинна триматися у таємниці користувачем A. A відправляє $(T_A^{'}, F_A^{'}, P_A^{'})$ і гарантію (w, θ) користувачу B через автентифікований канал, де $w = (pk_A, pk_B, t)$, t – це період часу, котрий відмічає що w діє протягом часу t, і θ це підпис на w, створений користувачем A з використанням алгоритму $Sign$, $\theta = Sign(H(w), sk_A)$.

Створення проксі ключів: Після отримання $(T_A^{'}, F_A^{'}, P_A^{'}, w, \theta)$, користувач B, передусім, перевіряє справжність $P_A^{'}$, θ, перевіряючи $P_A^{'} = F_A^{'} \circ T_A^{'}$ і $Verify(w, \theta, pk_A) = 1$. Потім B обирає випадкове бієктивне афінне відображення $T^{'}$ і обчислює $T_p = T^{'-1} \circ T_A^{'}$ і $F_p = F_A^{'} \circ T^{'}$. Нехай sk_p – це секретний для звичайного підпису і, відповідно, відкритий ключ

– це pk_p, це тому що наступні рівності виконуються: $F_p \circ T_p = F_A^{'} \circ T^{'} \circ T^{'-1} \circ T \circ T_A = F_A^{'} \circ T \circ T_A = F_A^{'} \circ T_A^{'} = P_A^{'}$. Потім B обчислює підпис $\sigma_{prx} = Sign(H(w, \theta, P_A^{'}), sk_B)$ і sk_p це проксі ключ, який B використовує для створення проксі підпису від особи користувача A і задає pk_p і $(w, \theta, P_A^{'}, \sigma_{prx})$ як проксі ключ перевірки.

Проксі підписування: Нехай M це повідомлення, яке потрібно підписати. Цей алгоритм створює проксі підпис на повідомленні M користувачем B. Користувач B використовує T_p і центральне відображення F_p у базовому алгоритмі підпису багатовимірної криптосистеми з відкритим ключем $Sign$, щоб створити підпис σ на повідомленні M.

$\sigma = Sign(H(M), sk_p)$

Потім проксі підпис на повідомленні M, створений користувачем B це $(\sigma, (w, \theta, P_A^{'}, \sigma_{prx}))$.

Проксі перевірка: На вході $(\sigma, (w, \theta, P_A^{'}, \sigma_{prx}))$ і будь-хто може перевірити справжність проксі підпису, виконуючи наступні кроки:

1) Перевіряємо правильність θ для гарантії w, використовуючи pk_A. $Verify((w, P_A^{'}), \theta, pk_A) \stackrel{?}{=} true$.

Якщо вони правильні, то переходимо до наступного кроку, інакше – виводимо 0.

2) Перевіряємо правильність σ_{prx} для $(w, \theta, P_A^{'})$, використовуючи pk_B. $Verify((w, \theta, P_A^{'}), \sigma_{prx}, pk_B) \stackrel{?}{=} true$.

Якщо вони правильні, то переходимо до наступного кроку, інакше – виводимо 0.

3) Перевіряємо правильність σ на повідомленні M, використовуючи $P_A^{'}$. $Verify(M, \sigma, P_A^{'}) \stackrel{?}{=} true$.

Якщо вони правильні, то виводимо 1, інакше – виводимо 0.

Перевіряючий приймає проксі підпис тільки якщо одночасно виконуються усі три умови, інакше – проксі підпис відхиляється.

2.4 Огляд загальної порогової проксі схеми шифропідпису у багатовимірній криптосистемі з відкритим ключем

Позначення: оригінальний підписувач – id_o, група проксі шифропідписувачів – $Pr = id_1, \ldots, id_{num}$, num – кількість проксі шифропідписувачів, отримувач – id_r.

Схема складається з 5ти алгоритмів [34]:

Задання початкових параметрів(1^λ)): Алгоритм ініціалізації системи, який виконується центром створення ключів(ЦСК). 1^λ) подається на вхід ЦСК і ЦСК обирає необхідний параметр $params$ для системи.

Створення($id, params, s$): Цей алгоритм включає в себе створення часткових відкритих і секретних ключів користувача і створення фінального ключа користувача.

Створення проксі ключів і гарантії: Цей алгоритм виконується оригінальним підписувачем id_o і головним чином створює проксі ключі для шифропідпису для проксі підписувачів і автентифікації інформації, яка використовується, щоб показати особу оригінального шифропідписувача і проксі шифропідписувачів, стан типів повідомлень і необхідної "авторизації".

Пороговий проксі шифропідпис: На вхід подаються відповідно $params$ і повідомлення m, проксі шифропідписувачі створять шифротекст як законні представники оригінального шифропідписувача.

Розшифровка та перевірка підпису: Цей алгоритм виконується отримувачем id_r, кінцевий отримувач може отримати повідомлення, якщо перевірка закінчиться успішно.

1. Задання початкових параметрів: Як вхід береться безпечний параметр 1^λ, ЦСК обирає деякі відповідні параметри згідно з секретним параметром і генерує інші параметри системи. По-перше, ЦСК обирає скінченне поле $\mathbb{F}^n$, p – генератор цього поля, порядком поля є

$q(q = p^k, k > 0)$. Обирається безпечна геш-функція $H_0 : \{0,1\}^{(n+1)l_{id}} \times \{0,1\}^* \to \mathbb{F}^n$, де l_{id} - це бітова довжина ідентифікатора id. $H_1 : \{0,1\}^* \to \mathbb{F}^n$, $H_2 : \{0,1\}^* \to \{0,1\}^{l_m}$, де l_m - це бітова довжина повідомлення m. Обираються багатовимірні квадратичні поліном, який можна обернути $F : \mathbb{F}^n \to \mathbb{F}^n$ і два афінні відображення, які можна обернути $T : \mathbb{F}^n \to \mathbb{F}^n$ і $S : \mathbb{F}^n \to \mathbb{F}^n$, потім обчислюється $F^\times = T \circ F \circ S$. Відкритий ключ ЦСК це $(F^\times, F)$, а секретний ключ системи це (T, S). ЦСК публікує $params = < G, p, q, H_0, H_1, H_2, F^\times >$ і зберігає в таємниці секретний ключ (T, S).

2. Створення: видобування ключів складається з двох алгоритмів. Маючи id користувача ЦСК створює частковий секретний ключ користувача, який користувач потім використовує, щоб обчислити свій фінальний відкритий ключ і секретний ключ.

2.1 Створення часткового відкритого і секретного ключів користувача: Маючи id користувача, ЦСК обирає два афінні відображення, які можна обернути T'_{idp} і S'_{idp} для цього користувача і обчислює $F_{idp} = T'_{idp} \circ F^\times \circ S'_{idp}$. Частковий відкритий ключ – це F_{idp}, а частковий секретний ключ – це $(T_{idp} = T'_{idp} \circ T, S_{idp} = S'_{idp} \circ S)$. ЦСК відправляє частковий секретний ключ користувачу через захищений канал.

2.2 Створення кінцевих відкритих і секретних ключів користувача: Після отримання часткового секретного ключа від ЦСК, користувач id випадковим чином обирає дві афінні відображення, які можна обернути (T'_{id}, S'_{id}) в скінченному полі і обчислює $F_{id} = T'_{id} \circ F_{idp} \circ S'_{id}$. Відкритий ключ користувача – це F_{id}, а секретний ключ користувача – це $(T_{id} = T'_{id} \circ T'_{idp} \circ T, S_{id} = S \circ S'_{idp} \circ S'_{id})$. Відкритий ключ публікується, секретний ключ зберігається в таємниці. У цьому випадку ЦСК не знає секретний ключ користувача. Користувач відправляє відкритий ключ до ЦСК, який буде його публікувати.

3. Створення проксі ключів і гарантії: Щоб реалізувати проксі шифропідпис, Аліса повинна авторизувати групу з num

шифропідписувачів, позначаючи $Pr = id_1, \ldots, id_{num}$, щоб зробити шифропідпис від її особи, так що Аліса виконує наступні кроки, щоб закінчити авторизацію і створення сертифіката "авторизації"m_w і ключа для проксі шифропідписа.

3.1 Створення проксі ключів і гарантії: Оригінальний шифропідписувач Аліса, створює її проксі ключ шифропідпису і ділиться ним з num проксі шифропідписувачами. Відкритий ключ Аліси – це F_{Alice} і її секретний ключ – це (T_{Alice}, S_{Alice}). Аліса обирає два афінні відображення, які можна обернути $(T_p^{'}, S_p^{'})$ нескінченного поля, і обчислює $T_p = T_p^{'} \circ T_{Alice}^{'}$ і $S_p = S_{Alice}^{'} \circ S_p^{'}$. Потім секретний ключ для проксі шифропідпису це (T_p, S_p). Відповідні відкриті ключі – це $F_p = T_p^{'} \circ T_{Alice}^{'} \circ F_{Alice} \circ S_{Alice}^{'} \circ S_p^{'} = T_p \circ F_{Alice} \circ S_p$. Операції обернення відкритого ключа проксі шифропідпису – це $F_p^{-1} = S_p^{-1} \circ F_{Alice}^{-1} \circ T_p^{-1}$. Потім Алісу делегує проксі ключ до num проксі шифропідписувачів. Обидві T_p і S_p – це афінні відображення, які можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$ і відповідні обернені відображення T_p^{-1} і S_p^{-1} також афінні відображення, які можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$. Тут $T_p^{-1} = (((T_p^{-1})^{(1)})^T, ((T_p^{-1})^{(2)})^T, \ldots, ((T_p^{-1})^{(n)})^T)^T$. Де $(T_p^{-1})_{i=1,2,\ldots,n}^{(i)}$ – це і-тий елемент T_p^{-1}.

3.2 Розподіл секрету: Аліса обирає num різних чисел $\beta_i \in \mathbb{F}_{i=1,2,\ldots,num}^n$ і використовує їх, щоб побудувати матрицю Вандермонда:

$$A = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \beta_1 & \beta_2 & \cdots & \beta_{num} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_1^{n-1} & \beta_2^{n-1} & \cdots & \beta_{num}^{n-1} \end{pmatrix}$$

Потім обчислюється наступне:

$$B1 = T_p^{-1} \times A = \begin{pmatrix} (T_p^{-1})^{(1)} \\ (T_p^{-1})^{(2)} \\ \vdots \\ (T_p^{-1})^{(n)} \end{pmatrix} \times A = \begin{pmatrix} b1_{1,1} & b1_{1,2} & \cdots & b1_{1,num} \\ b1_{2,1} & b1_{2,2} & \cdots & b1_{2,num} \\ \vdots & & & \\ b1_{n,1} & b1_{n,2} & \cdots & b1_{n,num} \end{pmatrix} =$$

$$= (B1[1], B1[2], \ldots, B1[num])$$

$$B2 = S_p^{-1} \times A = \begin{pmatrix} (S_p^{-1})^{(1)} \\ (S_p^{-1})^{(2)} \\ \vdots \\ (S_p^{-1})^{(n)} \end{pmatrix} \times A = \begin{pmatrix} b2_{1,1} & b2_{1,2} & \cdots & b2_{1,num} \\ b2_{2,1} & b2_{2,2} & \cdots & b2_{2,num} \\ \vdots & & & \\ b2_{n,1} & b2_{n,2} & \cdots & b2_{n,num} \end{pmatrix} =$$

$$= (B2[1], B2[2], \ldots, B2[num])$$

$(B1[i], B2[i])_{i=1,2,\ldots,num}$ представляють і-ий стовпчик вектора. Тому Аліса повинна розділити проксі ключ на num частин та роздати їх. Аліса робить $B_{id_i} = (B1[i], B2[i])$ як секретний ключ проксі шифропідписувача id_i і відправляє його проксі шифропідписувачу через захищений канал.

3.3 Формування гарантії: Аліса відправляє (B_{id_i}, β_i) кожному проксі шифропідписувачу id_i, де $(i = 1, 2, \ldots, num)$, через захищений канал. Аліса створює "авторизацію"m_w, яка складається з ідентифікатора оригінального підписувача id_{Alice}, членів групи проксі шифропідписувачів $Pr = id_1, \ldots, id_{num}$, назначену законну дату вигасання проксі шифропідпису t, відкритих ключів проксі шифропідписувачів і деякої необхідної інформації, $m_w =< id_{Alice}, id_1, \ldots, id_{num}, t, F_p >$. Потім Аліса підписує "авторизацію використовуючи геш-функцію H_0 і обчислює War за допомогою оберненого відкритого ключа Аліси F_{Alice}^{-1}, $War = F_{Alice}^{-1}(H_0(m_w))$, Аліса публікує (m_w, War).

4. Пороговий проксі шифропідпис: Припускаємо, що існує принаймні n проксі шифропідписувачів, які співпрацюють для шифропідписування повідомлення m для Аліси в пороговій схемі проксі

шифропідпису.

4.1 Автентифікація осіб проксі шифропідписувачів: Передусім, кожен проксі шифропідписувач повідомляє свій ідентифікатор іншим $n-1$ проксі шифропідписувачам. За допомогою запиту відкритої "авторизації"m_w, кожен проксі шифропідписувач переконується, що ідентифікатори інших проксі шифропідписувачів знаходяться в тій самій групі. Потім кожен проксі шифропідписувач перевіряє $F_{Alice}(War) = H_0(m_w)$. Якщо ця рівність не виконується, створення шифропідпису відхиляється, інакше – продовжується виконання наступних кроків. Якщо особи усіх інших проксі шифропідписувачів законні, наступний крок виконується, інакше – алгоритм переривається. Через минулі операції кожен учасник може отримати список ідентифікаторів $list =< id_{j_1}, id_{j_2}, \ldots, id_{j_n} >$, де $1 \leqslant j_1 < j_2 < \ldots < j_n \leqslant num$.

4.2 Розподіл секрету і створення шифропідпису: Відповідно до порядку списку ідентифікаторів, перший проксі шифропідписувач з n проксі шифропідписувачів випадковим чином обирає значення $r = \mathbb{F}^n$, потім обчислює $R = F^{\times}(r)$ і $Y = H_1(m||R||id_{Alice}||list)$.

1. Кожен секретний ключ проксі шифропідписувача це $B_{id_{j_i}} = (B1[j_i], B2[j_i])_{i=1,2,\ldots,n}$. Відповідно до порядку в списку ідентифікаторів, кожен проксі шифропідписувач id_{j_i} відправляє k-ий елемент $(B1[j_i]_k, B2[j_i]_k)$ його секретного ключа і β_{j_i} користувачу id_{j_k}. Потім кожен проксі шифропідписувач id_{j_i} виконує наступні обчислення:

$$(B1[j_1]_i, B1[j_2]_i, \ldots, B1[j_n]_i) \times A^{-1} = (PV1_{i1}, PV1_{i2}, \ldots, PV1_{in}) = PV1_i,$$

$$(B2[j_1]_i, B2[j_2]_i, \ldots, B2[j_n]_i) \times A^{-1} = (PV2_{i1}, PV2_{i2}, \ldots, PV2_{in}) = PV2_i.$$

2. Кожен проксі шифропідписувач id_{j_i} обчислює $y_i = PV1_i(Y)$для довіреної третьої сторони. Отримуючи y_i від n проксі шифропідписувачів, ЦСК обчислює $Y^{'} = (y_1, \ldots, y_n)$ і $Y^{''} = F^{-1}_{Alice}(Y^{'})$, використовуючи частковий секретний ключ оригінального підписувача. Потім ЦСК повідомляє $Y^{''}$ n проксі шифропідписувачам.

3. Після отримання $Y^{''}$ кожен проксі шифропідписувач id_{j_i}

обчислює $l_i = PV2_i(Y'')$ і $sig_i = F^{-1}_{id_{j_i}}(Y)$ і повідомляє (l_i, sig_i) іншим проксі шифропіписувачам.

4. Кожен проксі шифропідписувач id_{j_i} збирає інформацію $L = (l_1, \ldots, l_n)$ і $Sig = (sig_1, \ldots, sig_n)$, потім обчислює $W_{j_i} = F_{receiver}(L||Sig||R)$ і $Z_{j_i} = H_2(L, Sig, R) \oplus m$.

5. Проксі шифропідписувачі відправляють $(W = W_{j_i}, Z = Z_{j_i}, list)$ до довіреної третьої сторони, котра перевіряє чи $(W, Z, list)$ створені кожним проксі шифропідписувачем відповідають своїм значенням. Якщо так, фінальний шифропідписаний шифротекст це $\sigma = (id_{Alice}, list, L, Z, W)$. Інакше – довірена третя сторона ігнорує набір $(W, Z, list)$. Наостанок, довірена третя сторона відправляє шифротекст отримувачу $F_{receiver}$.

5. Розшифровка і перевірка: Коли отримувач отримав пороговий проксі шифропідписане повідомлення $\sigma = (id_{Alice}, list, L, Z, W)$, він буде виконувати алгоритм розшифровки. По-перше, він перевірить чи ідентифікатори проксі шифропідписувачів зі списку законні за допомогою запиту відкритої інформації з "авторизації"таким чином: $F_{Alice}(War) = H_0(m_w)$, n запитів чи порогові проксі шифропідписувачі мають проксі властивість протягом періоду ефективності. Якщо результати перевірки позитивні, отримувач виконує наступні кроки:

1. Використовуючи свій власний секретний ключ, отримувач може обчислити $F^{-1}_{receiver}(W) = L^{'}||Sig^{'}||R^{'}$ і $Sig^{'} = (sig^{'}_1, \ldots, sig^{'}_n)$.

2.Оцінюємо $L = L^{'}$. Якщо ця рівність не виконується, отримувач відхиляє шифропідписаний шифротекст. Якщо виконується, отримувач продовжує обчислювати $m^{'} = Z \oplus H_2(L^{'}||Sig^{'}||R^{'})$ і перевіряє чи тип повідомлення авторизований. Якщо ні, алгоритм зупиняється і повертається 0.

3. Отримувач перевіряє чи наступні рівності виконуються

$F_p(L) = H_1(m^{'}||R^{'}||id_{Alice}||list)$

$= F_{id_1}(sig^{'}_1) = \ldots = F_{id_n}(sig^{'}_n)$.

Якщо рівності подані вище виконуються, отримувач дозволяє

шифротекст і відкритий текст як m'. Інакше - відхиляє шифротекст і повертає 0.

Висновки до розділу 2

У розділі досліджено актуальність використання цифрових підписів зі спеціальними властивостями. Визначено поняття проксі підпису та типи делегування у такого типу підписах. Розглянуто порогові підписи та особливості їх використання. Проведено огляд схем підписів, які поєднують у собі порогові, проксі та сліпі підписи, а також досліджено методи побудови таких підписів. Визначено основні параметри, які впливають на захищеність подібних схем цифрового підпису.

3 ПОБУДОВА ПОРОГОВОЇ ПРОКСІ СХЕМИ ЦИФРОВОГО ПІДПИСУ У БАГАТОВИМІРНІЙ КРИПТОСИСТЕМІ З ВІДКРИТИМ КЛЮЧЕМ

Оскільки порогова схема забезпечує можливість підписання документу певною підгрупою групи підписувачів, а властивість проксі дає змогу передати здібність підписування документів, то порогова проксі схема цифрового підпису насамперед актуальна для великих компаній, де є рада директорів, яка у відсутність голови компанії повинна приймати рішення про прийняття того чи іншого рішення і відповідно підписування якогось документу.

З метою зниження кількості використаних ресурсів для створення підпису була обрана схема цифрового підпису LUOV, так як саме ця схема створена виключно для побудови цифрових підписів, та загалом, за рахунок правильного підбору параметрів та реалізації арифметики у скінченних полях, дає змогу значно пришвидшити процес створення цифрового підпису.

Такі схеми цифрового підпису базуються на ідентифікаторах, тому оригінільний підписувач завжди буде знати, хто прийняв рішення про підписування та створив цифровий підпис, такой наступні схеми мають назначеного перевіряючого, у такій проксі підпис може бути перевірений тільки перевіряючим назначеними оригінальним підписувачем, це дає змогу забезпечити автентифікацію повідомлень без відмови. До того лише назначений перевіряючий може перевірити правильність підпису і не може переконати в цьому когось іншого.

3.1 Загальна порогова проксі схема цифрового підпису у багатовимірній криптосистемі з відкритим ключем

Задання початкових параметрів: 1^λ – захищений параметр, ЦСК вибирає скінченне поле $\mathbb{F}^n$, p – генератор цього поля, порядком поля є $q(q = p^k, k > 0)$, захищені геш-функції $H_0 : \{0,1\}^{(n+1)l_{id}} \times \{0,1\}^* \to \mathbb{F}^n$, де l_{id} - це бітова довжина ідентифікатора id, $H_1 : \{0,1\}^* \to \mathbb{F}^n$, багатовимірний квадратичний поліном, який можна обернути $F : \mathbb{F}^n \to \mathbb{F}^n$ і два афінні відображення, які можна обернути $T : \mathbb{F}^n \to \mathbb{F}^n$ і $S : \mathbb{F}^n \to \mathbb{F}^n$. Обчислюється $F^\times = T \circ F \circ S$. Відкритий ключ ЦСК це $(F^\times, F)$, а секретний ключ системи це (T, S). ЦСК публікує $params = < G, p, q, H_0, H_1, F^\times >$.

Створення ключів: Маючи id користувача, ЦСК вибирає два афінні відображення, які можна обернути T'_{idp} і S'_{idp} для цього користувача і обчислює $F_{idp} = T'_{idp} \circ F^\times \circ S'_{idp}$. Частковий відкритий ключ – це F_{idp}, а частковий секретний ключ – це $(T_{idp} = T'_{idp} \circ T, S_{idp} = S'_{idp} \circ S)$. ЦСК відправляє частковий секретний ключ користувачу через захищений канал.

Після отримання часткового секретного ключа, користувач id випадковим чином вибирає дві афінні відображення, які можна обернути (T'_{id}, S'_{id}) в скінченному полі і обчислює $F_{id} = T'_{id} \circ F_{idp} \circ S'_{id}$. Відкритий ключ користувача – це F_{id}, а секретний ключ користувача – це $(T_{id} = T'_{id} \circ T'_{idp} \circ T, S_{id} = S \circ S'_{idp} \circ S'_{id})$.

Створення проксі ключів: Оригінальний підписувач Аліса, створює свій проксі ключ підпису і ділиться ним з num проксі підписувачами. Відкритий ключ Аліси – це F_{Alice} і її секретний ключ – це (T_{Alice}, S_{Alice}). Аліса вибирає два афінні відображення, які можна обернути (T'_p, S'_p) нескінченного поля, і обчислює $T_p = T'_p \circ T'_{Alice}$ і $S_p = S'_{Alice} \circ S'_p$. Потім секретний ключ для проксі підпису це (T_p, S_p). Відповідні відкриті ключі – це

$F_p = T_p^{'} \circ T_{Alice}^{'} \circ F_{Alicep} \circ S_{Alice}^{'} \circ S_p^{'} = T_p \circ F_{Alicep} \circ S_p$. Операції обернення відкритого ключа проксі підпису – це $F_p^{-1} = S_p^{-1} \circ F_{Alicep}^{-1} \circ T_p^{-1}$. Потім Алісу делегує проксі ключ до num проксі підписувачів. Обидві T_p і S_p – це афінні відображення, які можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$ і відповідні обернені відображення T_p^{-1} і S_p^{-1} також афінні відображення, які можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$. Тут $T_p^{-1} = (((T_p^{-1})^{(1)})^T, ((T_p^{-1})^{(2)})^T, \ldots, ((T_p^{-1})^{(n)})^T)^T$. Де $(T_p^{-1})_{i=1,2,\ldots,n}^{(i)}$ – це і-тий елемент T_p^{-1}.

Делегування: Аліса вибирає num різних чисел $\beta_i \in \mathbb{F}_{i=1,2,\ldots,num}^n$ і використовує їх, щоб побудувати матрицю Вандермонда:

$$A = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \beta_1 & \beta_2 & \cdots & \beta_{num} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_1^{n-1} & \beta_2^{n-1} & \cdots & \beta_{num}^{n-1} \end{pmatrix}$$

Потім обчислюється наступне:

$$B1 = T_p^{-1} \times A = \begin{pmatrix} (T_p^{-1})^{(1)} \\ (T_p^{-1})^{(2)} \\ \vdots \\ (T_p^{-1})^{(n)} \end{pmatrix} \times A = \begin{pmatrix} b1_{1,1} & b1_{1,2} & \cdots & b1_{1,num} \\ b1_{2,1} & b1_{2,2} & \cdots & b1_{2,num} \\ \vdots & & & \\ b1_{n,1} & b1_{n,2} & \cdots & b1_{n,num} \end{pmatrix} =$$

$$= (B1[1], B1[2], \ldots, B1[num])$$

$$B2 = S_p^{-1} \times A = \begin{pmatrix} (S_p^{-1})^{(1)} \\ (S_p^{-1})^{(2)} \\ \vdots \\ (S_p^{-1})^{(n)} \end{pmatrix} \times A = \begin{pmatrix} b2_{1,1} & b2_{1,2} & \cdots & b2_{1,num} \\ b2_{2,1} & b2_{2,2} & \cdots & b2_{2,num} \\ \vdots & & & \\ b2_{n,1} & b2_{n,2} & \cdots & b2_{n,num} \end{pmatrix} =$$

$$= (B2[1], B2[2], \ldots, B2[num])$$

$(B1[i], B2[i])_{i=1,2,\ldots,num}$ представляють і-ий стовпчик вектора. Тому Аліса повинна розділити проксі ключ на num частин та роздати їх. Аліса робить $B_{id_i} = (B1[i], B2[i])$ як секретний ключ проксі підписувача id_i і відправляє його проксі підписувачу через захищений канал.

Створення гарантії Аліса відправляє (B_{id_i}, β_i) кожному проксі підписувачу id_i, де $(i = 1, 2, \ldots, num)$, через захищений канал. Аліса створює "гарантію"m_w, яка складається з ідентифікатора оригінального підписувача id_{Alice}, членів групи проксі підписувачів $Pr = id_1, \ldots, id_{num}$, назначену законну дату t, коли проксі підпис втратить силу, відкритих ключів проксі підписувачів і деякої необхідної інформації, $m_w =< id_{Alice}, id_1, \ldots, id_{num}, t, F_p >$. Потім Аліса підписує "гарантію використовуючи геш-функцію H_0 і обчислює War за допомогою оберненого відкритого ключа Аліси F^{-1}_{Alice}, $War = F^{-1}_{Alice}(H_0(m_w))$, Аліса публікує (m_w, War). Кожен проксі підписувач повідомляє свій ідентифікатор іншим $n - 1$ проксі підписувачам. За допомогою запиту відкритої "гарантії"m_w, кожен проксі підписувач переконується, що ідентифікатори інших проксі підписувачів знаходяться в тій самій групі. Потім кожен проксі підписувач перевіряє $F_{Alice}(War) = H_0(m_w)$. Якщо ця рівність не виконується, створення підпису відхиляється, інакше – продовжується виконання наступних кроків. Якщо особи усіх інших проксі підписувачів законні, наступний крок виконується, інакше – алгоритм переривається. Через минулі операції кожен учасник може отримати список ідентифікаторів $list =< id_{j_1}, id_{j_2}, \ldots, id_{j_n} >$, де $1 \leqslant j_1 < j_2 < \ldots < j_n \leqslant num$.

Проксі підписування: Відповідно до порядку списку ідентифікаторів, перший проксі підписувач з n проксі підписувачів випадковим чином вибирає значення $r_i = \mathbb{F}^n$, потім обчислює $R_i = F^{\times}(r_i)$ і $Y = H_1(m||R_i||id_{Alice}||list)$.

1. Кожен секретний ключ проксі підписувача це $B_{id_{j_i}} = (B1[j_i], B2[j_i])_{i=1,2,\ldots,n}$. Відповідно до порядку в списку ідентифікаторів, кожен проксі підписувач id_{j_i} відправляє k-ий елемент

$(B1[j_i]_k, B2[j_i]_k)$ його секретного ключа і β_{j_i} користувачу id_{j_k}. Потім кожен проксі підписувач id_{j_i} виконує наступні обчислення:

$(B1[j_1]_i, B1[j_2]_i, \ldots, B1[j_n]_i) \times A^{-1} = (PV1_{i1}, PV1_{i2}, \ldots, PV1_{in}) = PV1_i,$

$(B2[j_1]_i, B2[j_2]_i, \ldots, B2[j_n]_i) \times A^{-1} = (PV2_{i1}, PV2_{i2}, \ldots, PV2_{in}) = PV2_i.$

2. Кожен проксі підписувач id_{j_i} обчислює $y_i = PV1_i(Y)$для довіреної третьої сторони. Отримуючи y_i від n проксі підписувачів, ЦСК обчислює $Y^{'} = (y_1, \ldots, y_n)$ і $Y^{''} = F^{-1}_{Alice}(Y^{'})$, використовуючи частковий секретний ключ оригінального підписувача. Потім ЦСК повідомляє $Y^{''}$ n проксі підписувачам.

3. Після отримання $Y^{''}$ кожен проксі підписувач id_{j_i} обчислює $l_i = PV2_i(Y^{''})$ і підпис $sig_i = F^{-1}_{id_{j_i}}(Y)$ і повідомляє (l_i, sig_i) іншим проксі піписувачам.

4. Кожен проксі підписувач id_{j_i} збирає інформацію $L = (l_1, \ldots, l_n)$ і загальний підпис $Sig = (sig_1, \ldots, sig_n)$, потім обчислює $W_{j_i} = F_{receiver}(L||Sig||R_i)$.

5. Проксі підписувачі відправляють $(W = W_{j_i}, list)$ до довіреної третьої сторони, котра перевіряє чи $(W, list)$ створені кожним проксі підписувачем відповідають своїм значенням. Якщо так, фінальний підписаний текст це $\sigma = (id_{Alice}, list, L, W)$. Інакше – довірена третя сторона ігнорує набір $(W, list)$. Наостанок, довірена третя сторона відправляє текст отримувачу $F_{receiver}$.

Перевірка: Коли отримувач отримав пороговий проксі підписане повідомлення $\sigma = (id_{Alice}, list, L, W)$, він буде виконувати алгоритм перевірки. По-перше, він перевірить чи ідентифікатори проксі підписувачів зі списку законні за допомогою запиту відкритої інформації з "гарантії"таким чином: $F_{Alice}(War) = H_0(m_w)$, n запитів чи порогові проксі підписувачі мають проксі властивість протягом періоду ефективності. Якщо результати перевірки позитивні, отримувач виконує наступні кроки:

1. Використовуючи свій власний секретний ключ, отримувач може обчислити $F^{-1}_{receiver}(W) = L^{'}||Sig^{'}||R^{'}$ і $Sig^{'} = (sig^{'}_1, \ldots, sig^{'}_n)$.

2.Оцінюємо $L = L^{'}$. Якщо ця рівність не виконується, отримувач відхиляє підпис. Якщо виконується, то переходить до наступного кроку.

3. Отримувач перевіряє чи наступні рівності виконуються

$F_p(L) = Y = H_1(m^{'}||R^{'}||id_{Alice}||list)$

$= F_{id_1}(sig_1^{'}) = \ldots = F_{id_n}(sig_n^{'})$.

Якщо рівності подані вище виконуються, отримувач приймає підпис Sig і повертає 1. Інакше – відхиляє і повертає 0.

3.2 Порогова проксі схема цифрового підпису на основі схеми підпису LUOV

На даний момент на другий етап конкурсу від NIST вийшли наступні схеми цифрових підписів багатовимірної криптографії: Rainbow, LUOV, GeMSS, MQDSS.

Схема LUOV (Lifted Unbalanced Oil and Vinegar) це покращена версія схеми цифрового підпису UOV (Unbalanced Oil and Vinegar) [35]. Ця схема відрізняється від UOV у трьох пунктах.

Перша модифікація, яку зробив Петцольд, це зміни в алгоритмі створення ключа, зробивши можливим вибір великої частини відкритого ключа. Модифікований алгоритм генерації ключів генерує розподіл відкритих поліноміальних відображень $\mathcal{P}$, і його не можна відрізнити від оригінальної схеми підпису, якщо ми приймемо вихід алгоритму PRNG, як справжню випадковість.

Другою модифікацією це те, що відкритий ключ $\mathcal{P} : \mathbb{F}_2^m \to \mathbb{F}_2^m$ для схеми UOV над полем $\mathbb{F}_2$, використовується як відкритий ключ схеми UOV над розширенням поля $\mathbb{F}_{2^r}$. Завдячуючи цьому відкритий ключ стає меншим.

Третя відмінність полягає у тому що лінійне відображення $\mathcal{T}$

вибирається з матричним представленням іншої форми $\begin{pmatrix} 1_v & T \\ 0 & 1_m \end{pmatrix}$, де T є матрицею розмірності $(v * m)$. Такий вибір робить процес створення ключа швидшим і не впливає на захищенність схеми.

Оновлена версія LUOV використовує скінченні поля $\mathbb{F}_{2^r}$, де r є простим. Це гарантує відсутність нетривіальних підполів, які можуть бути використані зловмисником. Більше того, реалізація арифметики поля в нових полях відбувається швидше, ніж у старих полях, що прискорює алгоритми підписання та перевірки. Вплив зміни полів на розмір ключа та підпису є мінімальним (ключі стали на кілька відсотків більше, а підписи – на кілька відсотків менше).

Схема UOV зроблена як схема підпису і не підходить для шифрування. Тому, приймаючи до уваги існування покращеної версії схеми цифрового підпису LUOV, яка дозволить зменшити кількість ресурсів, які будуть витрачатися на створення порогового проксі підпису, прийнято рішення адаптації порогового проксі підпису для схеми LUOV.

Задання початкових параметрів: На вході захищений параметр 1^λ. ЦСК вибирає скінченне поле $\mathbb{F}^n$, p – генератор цього поля, порядком поля є $q(q = p^k, k > 0)$, захищені геш-функції $H_0 : \{0,1\}^{(n+1)l_{id}} \times \{0,1\}^* \to \mathbb{F}^n$, де l_{id} - це бітова довжина ідентифікатора id, багатовимірний квадратичний поліном, який можна обернути $F : \mathbb{F}^n \to \mathbb{F}^n$ і афінне відображення, яке можна обернути $T : \mathbb{F}^n \to \mathbb{F}^n$. Потім обчислюється $F^\times = F \circ T$. Відкритий ключ ЦСК це $F^\times$, а секретний ключ системи це T. ЦСК публікує $params = < G, p, q, H_0, F^\times >$.

Створення ключів: ЦСК вибирає афінне відображення, яке можна обернути T'_{idp} для користувача id і обчислює $F_{idp} = F^\times \circ T'_{idp}$ і $T_{idp} = T'_{idp} \circ T$. Частковий відкритий ключ – це F_{idp}, а частковий секретний ключ – це T_{idp}. ЦСК відправляє частковий секретний ключ користувачу через захищений канал. Після отримання часткового

секретного ключа, користувач id випадковим чином вибирає афінне відображення, яке можна обернути T'_{id} в скінченному полі і обчислює $F_{id} = F_{idp} \circ T'_{id}$. Відкритий ключ користувача – це F_{id}, а секретний ключ користувача – це T_{id}.

Створення проксі ключів: Оригінальний підписувач Аліса, створює свій проксі ключ підпису і ділиться ним з num проксі підписувачами. Відкритий ключ Аліси – це F_{Alice} і її секретний ключ – це T_{Alice}. Аліса вибирає афінне відображення, яке можна обернути T'_p нескінченного поля, і обчислює $T_p = T'_p \circ T'_{Alice}$. Потім секретний ключ для проксі підпису – це T_p. Відповідні відкриті ключі – це $F_p = F_{Alicep} \circ T'_p \circ T'_{Alice} = F_{Alicep} \circ T_p$. Операції обернення відкритого ключа проксі підпису – це $F_p^{-1} = T_p^{-1} \circ F_{Alicep}^{-1}$. Потім Аліса делегує проксі ключ до num проксі підписувачів. T_p – це афінне відображення, яке можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$ і відповідне обернене відображення T_p^{-1} також афінне відображення, які можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$. Тут $T_p^{-1} = (((T_p^{-1})^{(1)})^T, ((T_p^{-1})^{(2)})^T, \ldots, ((T_p^{-1})^{(n)})^T)^T$. Де $(T_p^{-1})^{(i)}_{i=1,2,\ldots,n}$ – це і-тий елемент T_p^{-1}.

Алгоритм 3.1. $ProxySecretKeyDelegation(T_p^{-1})$

Вхід: T_p^{-1} – обернене відображення для проксі підпису

Вихід: $B_{id} = (B_{id_1}, \ldots, B_{id_{num}})$ – секретні ключі проксі підписувачів

Початок

0: Вибирається num різних чисел $\beta_i \in \mathbb{F}^n_{i=1,2,\ldots,num}$.

1: Будується матриця Вандермонда $A = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \beta_1 & \beta_2 & \cdots & \beta_{num} \\ \vdots & \vdots & \vdots & \vdots \\ \beta_1^{n-1} & \beta_2^{n-1} & \cdots & \beta_{num}^{n-1} \end{pmatrix}$.

2: Обчислюється $B = T_p^{-1} \times A = \begin{pmatrix} (T_p^{-1})^{(1)} \\ (T_p^{-1})^{(2)} \\ \vdots \\ (T_p^{-1})^{(n)} \end{pmatrix} \times A =$

$$= \begin{pmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,num} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,num} \\ \vdots & & & \\ b_{n,1} & b_{n,2} & \cdots & b_{n,num} \end{pmatrix} = (B[1], B[2], \ldots, B[num]).$$

3: Береться і-ий стовпчик ветора $B[i]_{i=1,2,\ldots,num}$ і утворюється $B_{id_i} = B[i]$ – секретний ключ підписувача id_i.

4: Повертається $B_{id} = (B_{id_1}, \ldots, B_{id_{num}})$ – секретні ключі проксі підписувачів.

Кінець

Делегування: Аліса бере афінне відображення T_p і за допомогою алгоритму $ProxySecretKeyDelegation$ утворює секретні ключі підписувачів $B_{id} = (B_{id_1}, \ldots, B_{id_{num}}) = ProxySecretKeyDelegation(T_p^{-1})$, потім Аліса відправляє кожному підписувачу id_i його секретний ключ проксі через захищений канал.

Створення гарантії: Аліса відправляє (B_{id_i}, β_i) кожному проксі підписувачу id_i, де $(i = 1, 2, \ldots, num)$, через захищений канал. Аліса створює "гарантію"m_w, яка складається з ідентифікатора оригінального підписувача id_{Alice}, членів групи проксі підписувачів $Pr = id_1, \ldots, id_{num}$, назначену законну дату t, коли проксі підпис втратить силу, відкритих ключів проксі підписувачів і деякої необхідної інформації, $m_w =< id_{Alice}, id_1, \ldots, id_{num}, t, F_p >$. Потім Аліса підписує "гарантію використовуючи геш-функцію H_0 і обчислює War за

допомогою оберненого відкритого ключа Аліси F^{-1}_{Alice}, $War = F^{-1}_{Alice}(H_0(m_w))$, Аліса публікує (m_w, War). Кожен проксі підписувач повідомляє свій ідентифікатор іншим $n-1$ проксі підписувачам. За допомогою запиту відкритої "гарантії"m_w, кожен проксі підписувач переконується, що ідентифікатори інших проксі підписувачів знаходяться в тій самій групі. Потім кожен проксі підписувач перевіряє $F_{Alice}(War) = H_0(m_w)$. Якщо ця рівність не виконується, створення підпису відхиляється, інакше – продовжується виконання наступних кроків. Якщо особи усіх інших проксі підписувачів законні, наступний крок виконується, інакше – алгоритм переривається. Через минулі операції кожен учасник може отримати список ідентифікаторів $list =< id_{j_1}, id_{j_2}, \ldots, id_{j_n} >$, де $1 \leqslant j_1 < j_2 < \ldots < j_n \leqslant num$.

Проксі підписування: Відповідно до порядку у списку ідентифікаторів, перший проксі підписувач з n проксі підписувачів випадковим чином вибирає значення $r = \mathbb{F}^n$, потім обчислює $R = F^{\times}(r)$ і $Y = H_1(m||R||id_{Alice}||list)$.

1. Кожен секретний ключ проксі підписувача це $B_{id_{j_i}} = B[j_i]_{i=1,2,\ldots,n}$. Відповідно до порядку в списку ідентифікаторів, кожен проксі підписувач id_{j_i} відправляє k-ий елемент $B1[j_i]_k$ його секретного ключа і β_{j_i} користувачу id_{j_k}. Потім кожен проксі підписувач id_{j_i} виконує наступні обчислення:

$$(B[j_1]_i, B[j_2]_i, \ldots, B[j_n]_i) \times A^{-1} = (PV_{i1}, PV_{i2}, \ldots, PV_{in}) = PV_i,$$

2. Кожен проксі підписувач id_{j_i} обчислює $y_i = PV_i(Y)$для довіреної третьої сторони. Отримуючи y_i від n проксі підписувачів, ЦСК обчислює $Y^{'} = (y_1, \ldots, y_n)$ і $Y^{''} = F^{-1}_{Alice}(Y^{'})$, використовуючи частковий секретний ключ оригінального підписувача. Потім ЦСК повідомляє $Y^{''}$ n проксі підписувачам.

3. Після отримання $Y^{''}$ кожен проксі підписувач id_{j_i} обчислює підпис $sig_i = F^{-1}_{id_{j_i}}(Y)$ і повідомляє sig_i іншим проксі піписувачам.

4. Кожен проксі підписувач id_{j_i} збирає загальний підпис $Sig = (sig_1, \ldots, sig_n)$, потім обчислює $W_{j_i} = F_{verifier}(Sig||R_i)$.

5. Проксі підписувачі відправляють $(W = W_{j_i}, list)$ до довіреної третьої сторони, котра перевіряє чи $(W, list)$ створені кожним проксі підписувачем відповідають своїм значенням. Якщо так, фінальний підписаний текст це $\sigma = (id_{Alice}, list, W)$. Інакше – довірена третя сторона ігнорує набір $(W, list)$. Наостанок, довірена третя сторона відправляє текст перевіряючому $F_{verifier}$.

Перевірка: Коли перевіряючий отримав порогове проксі підписане повідомлення $\sigma = (id_{Alice}, list, W)$, він буде виконувати алгоритм перевірки. По-перше, він перевірить чи ідентифікатори проксі підписувачів зі списку законні за допомогою запиту відкритої інформації з "гарантії"таким чином: $F_{Alice}(War) = H_0(m_w)$, n запитів чи порогові проксі підписувачі мають проксі властивість протягом періоду ефективності. Якщо результати перевірки позитивні, отримувач виконує наступні кроки:

1. Використовуючи свій власний секретний ключ, перевіряючий може обчислити $F^{-1}_{verifier}(W) = Sig^{'}||R^{'}$ і $Sig^{'} = (sig^{'}_1, \ldots, sig^{'}_n)$.

2. Перевіряючий перевіряє чи наступні рівності виконуються

$F_p(Y^{''}) = Y = H_1(m^{'}||R^{'}||id_{Alice}||list)$

$= F_{id_1}(sig^{'}_1) = \ldots = F_{id_n}(sig^{'}_n)$.

Якщо рівності подані вище виконуються, отримувач приймає підпис Sig і повертає 1. Інакше – відхиляє і повертає 0.

3.3 Аналіз коректності і захищенності

Аналіз коректності

Теорема 3.1. *Процес перевірки у загальній пороговій проксі схемі цифрового підпису є коректним.*

Доведення. Після отримання підпису $\sigma = (id_{Alice}, list, L, W)$, перевіряючий виконує алгоритм перевірки. Перевіряючий використовує свій секретний ключ і обчислює

$F_{verifier}^{-1} = L^{'}||Sig^{'}||R^{'} = F_{verifier}^{-1}(F_{verifier}(L||Sig||R)) = L||Sig||R$. $L = L^{'}$ виконується для рівнянь поданих вище. Перевіряючий обчислює підпис за допомогою свого секретного ключа

$F_p(L) = Y = H_1(m^{'}||R^{'}||id_{Alice}||list) = H_1(m||R||id_{Alice}||list)$

$= F_{id_{j_t}}(sig_{j_i}^{'})$

$= F_{id_{j_t}}(Sig_{j_i}^{'}), (i = 1, \ldots, n).$

Тому обчислювальний процес перевірки є коректним. □

Доведення захищенності Багатовимірні криптосистеми з відкритим ключем головним чином базуються на складності MQ-задачі і IP-задачі.

MQ (Multivariate Quadratic)-задача: Є група рівнянь над скінченним полем і потрібно знайти набір змінних $x^- \in \mathbb{F}^n$, $x^- = (x_1^-, \ldots, x_n^-)$, такі що результуючі значення усіх рівнянь є 0, $f^{(1)}(x^-) = \ldots = f^{(m)}(x^-) = 0$. Знаходження $x^- = (x_1^-, \ldots, x_n^-)$, які відповідають усім вимогам, поданим вище, називається MQ-задачею.

IP (Isomorphism Polynomial)-задача: F і F^- два відкритих набори n рівнянь з n змінних над (p), якщо F і F^- це ізоморфізм, тоді $F^- = T \circ F \circ S$ ($\circ$ позначає композицію відображень), де T і S це два афінних відображення, які можна обернути над $\mathbb{F}^n \to \mathbb{F}^n$. Знаходження (T, S) з F і F^-, таких що $F^- = T \circ F \circ S$ називається IP-задачею.

Теорема 3.2. *Порогова проксі схема цифрового підпису на основі LUOV(TPMPKC-LUOV) є захищеною.*

Доведення. Оскільки схема цифрового підпису LUOV є стійкою, то TPMPKC-LUOV є також стійкою оскільки для її побудови використовувалося делегування з гарантією, яке не змінює саму схему цифрового підпису. Тому усі властивості оригінальної схеми цифрового підпису зберігаються, включно із захищеністю. □

Означення 3.1. Стійкість загальної порогової проксі схеми цифрового підпису у MPKC (TPMPKC) проти атак з обраним повідомлення і з обраною гарантією.

Нехай Q це $TPMPKC$. Наступні ігри граються між зловмисником F і претендентом C.

Задання початкових параметрів: C запускає цей алгоритм, щоб створити параметри системи $params$ і майстер-ключ. Він відсилає $params$ до фальсифікатора F і F виводить ідентифікатор цілі id_t.

Атака: Зловмисник може отримати доступ до будь-якого випадкового оракулу з поданих нижче:

1. Запит на створення секретного ключа: на вході маємо id користувача, випадковий оракул повертає на виході повний секретний ключ користувача PS.

2. Запит на створення відкритого ключа: на вході маємо id користувача, випадковий оракул повертає відкритий ключ користувача PK.

3. Запит на заміну відкритого ключа: на вході маємо id користувача і дійсний відкритий ключ PK, випадковий оракул замінює PK на PK$'$.

4. Запит на створення проксі гарантії: на вході маємо id_o оригінального підписувача і ідентифікатори проксі підписувачів $Pr = (id_1, \ldots, id_{num})$, випадковий оракул запускає алгоритм створення ключів і гарантії і повертає проксі ключ і гарантію m_w.

5. Запит на пороговий проксі підпис: на вході маємо повідомлення m, ідентифікатор оригінального підписувача id_o і проксі підписувачів $Pr = (id_1, \ldots, id_{num})$, випадковий оракул запускає алгоритм порогового проксі підпису і поветає підпис.

6. Порогова перевірка: на вході маємо підпис σ, ідентифікатор id_o оригінального підписувача і перевіряючого id_v, випадковий оракул виконує алгоритм перевірки і повертає 1 або 0.

Потрібно відмітити, що фальсифікатор не може виконати запит на видачу ключів і на заміну відкритого ключа для обраної цілі.

Підміна: F наприкінці досягає наступнрї мети підробки.

Зловмисник виводить підпис σ повідомлення m як підпис, який може бути успішно перевірений перевіряючим. Для створення підпису обраний

користувач id_t з проксі підписувачів. Також потребується, щоб id_t не був авторизованим оригінальним підписувачем і не видавав запит пороговий проксі підпис для повідомлення m. Такий тип атаки називається атака з обраним повідомленням.

Зловмисник виводить підпис σ повідомлення m як підпис, який може бути успішно перевірений перевіряючим. Для створення підпису обраний користувач id_t з проксі підписувачів і проксі підписувачі не повинні бути авторизовані оригінальним підписувачем. Такий тип атаки називається атака з обраною гарантією.

Перевага зловмисника F визначається як $Adv_Q^{3.1}$, що є ймовірністю виграти гру. Якщо перевага зловмисника у вищевказаній грі дуже незначна, схема називається стійкою.

Теорема 3.3. *Делегування у схемі TPMPKC є захищенним.*

Доведення. Оскільки процес делегування відбувається з використанням ідентифікаторів проксі підписувачів, на кожному етапі створення проксі підпису користувачі перевіряють відповідність даних за допомогою гарантії, яка містить необхідний набір даних для автентифікації усіх проксі підписувачів та їх проксі ключів. □

Теорема 3.4. *Для схеми TPMPKC, якщо є можливість, що за поліноміальний час зловмисник A може виграти гру наведену у визначенні 3.1 з незначною перевагою ϵ, також існує претендент C, який може перетворити можливість зловмисника A у перевагу у вирішенні IP-задачі за поліноміальний час. Ця перевага ϵ' задовольняє умову $\epsilon' \geqslant \frac{1}{q_{sc}}(1 - \frac{q_{ver}}{n2^{|G|}})$. У цій грі A може зробити щонайбільше $q_{H_i(i=0,1)}$ запитів геш-функцій $H_i(i = 0,1)$, кількість запитів на пороговий проксі підпис – це q_{sc}, запит на отримання відкритого проксі ключа – це q_{pke}, запит на отримання секретного ключа – це q_{ske}, а запит на перевірку – це q_{ver}.*

Доведення. Претендент C хоче вирішити екземпляр IP-задачі $(F^* = T^* \circ T_0 \circ F^- \circ S_0 \circ S^*, T_0 \circ F^- \circ S_0)$. У наступній грі C використовує

здібність A, щоб вирішити IP-задачу.

Задання початкових параметрів: C виконує цей алгоритм і задає параметри системи. G – скінченне поле, p – генератор цього поля, порядком поля є $q(q = p^k, k > 0)$. Випадковим чином вибираються два афінні відображення, які можна обернути (T, S) як секретний ключ системи. Задається відкритий ключ як $(F^- = T \circ F \circ S)$. C випадковим чином вибирає $T_0 : \mathbb{F}^n \to \mathbb{F}^n$ і $S_0 : \mathbb{F}^n \to \mathbb{F}^n$ і частковий секретний ключ системи – це $(T_0 \circ T, S \circ S_0)$, C відправляє $params =< G, p, q, H_0, H_1, F^- >$ до A. Фальсифікатор A отримує параметри системи і виводить ідентифікатор цілі id^* і відповідно його відкритий ключ – це F^*. З метою зробити запит до випадкового оракула $H_i(i = 0{,}1)$ від A, C зберігає кожен результат запитів у відповідний список $H_i - list(i = 0{,}1)$.

Фаза 1: A може виконати запит до випадкового оракула з визначення 3.1.

Запит перевірки: на вході маємо ідентифікатор перевіряючого id_v і підпис $\sigma = (id_s, list, L, W)$. Якщо $id_v \neq id^*$, C отримує секретний ключ за допомогою запиту секретного ключа, і передає його користувачу A. Інакше, C перевіряє чи $H_1 - list$ містить $(m, R, id_i, list, h1_i, T_p^{'}, S_p^{'})$. Якщо $F_p(L) = h1_i$ і рівність не виконується, то підпис незаконний. У цій фазі ймовірність провалу прийняття законного підпису як незаконного складає $\frac{q_v er}{n2^{|G|-1}}$. Фактично, ймовірність не знайти відповідний набір у $H_1 - list$ менше ніж $\frac{1}{n2^{|G|}}$. До цього ж q_{ver} запит перевірки, так що ймовірність відхилити правильний підпис складає менше ніж $\frac{q_v er}{n2^{|G|-1}}$.

Фаза підміни: після виконання поданого вище запиту до випадкового оракулу, A повертає підмінений підпис $(id_s, list, L^*, W^*)$ і підпис повідомлення m не може бути отриманий виконанням запитів. У підписі $id_s = id^*$, це означає що оригінальний підписувач це id^*. У цій ситуації потрібно, щоб id_s не можна було запитати за допомогою запиту на створення секретного ключа у поданих вище запитах і $(id_s, list)$ не може бути запитаний за допомогою запиту на проксі авторизацію. Якщо $id^* \in list$, потрібно, щоб перед поверненням підміненого підпису id_s не міг

запитати за допомогою запиту на створення секретного ключа і $(id_s, list)$ не могло бути запитаним за допомогою запиту гарантії і минулих запитів випадкового оракула.

За допомогою поданої вище гри можна зробити висновок, що процес ефективно симулює ситуації з атаками у реальності. Була проаналізована перевага C. Нехай E позначає подію, коли A виводить підмінений підпис успішно. Події E_1, E_2 і E_3 визначені таким чином: подія E_1 видає запит на ідентифікатор цілі, коли виконується запит на створення секретного ключа; подія E_2 позначає помилку підписування, оскільки перевіряючий є вибраною ціллю і деякому запиту на підписування; подія E_3 позначає помилку перевірки підпису на справжність, а C відхиляє справжній підпис. Коли відбувається подія E, E_1 і E_2 не відбуваються, це $\neg E_1 \wedge \neg E_2$. Ймовірність що відбудеться E_3 менше ніж $\frac{q_{ver}}{n2^{|G|}}$. E_4 позначає ймовірність того, що C вибере правильне значення з $H_0 - list$ під час фази вгадування і $Pr(E_4) \leqslant \frac{1}{q_{H_0}+q_{sc}}$. Перевага C – це $\epsilon^{'} = Pr(E \wedge \neg E_1 \wedge \neg E_2 \wedge \neg E_3 E_4)$, де $\epsilon^{'} \geqslant \frac{1}{q_{H_0}+q_{sc}}(1 - \frac{q_{ver}}{n2^{|G|}})$. □

Висновки до розділу 3

У цьому розділі побудовано дві порогові проксі схеми цифрового підпису. Перша схема є загальною для усіх схем цифрових підписів на основі багатовимірних криптосистем з відкритим ключем, у ній використовуються ідентифікатори і є назначений перевіряльник, який забезпечує автентифікацію без відмови. На її основі у подальшому можна будувати інші схеми порогових проксі підписів з використанням примітивів багатовимірної криптографії. А друга схема є варіацією для схеми цифрового підпису LUOV. Так як за рахунок правильного підбору параметрів та реалізації арифметики у скінченних полях, використання порогової проксі схеми на основі LUOV пришвидшує процес створення цифрового підпису, ця схема є оптимальною для використання, коли до цього процесу залучена велика група людей.

ВИСНОВКИ

В області цифрових підписів існує багато практичних та безпечних схем, однак бракує схем підписів зі спеціальними властивостями, такими як сліпі, порогові та проксі підписи. Враховуючи те, що у сучасному світі існує багато випадків, коли потрібно використовувати цифрові підписи зі спеціальними властивостями, наприклад, використання сліпих підписів для анонімної ідентифікації, або використання проксі підписів у ситуаціях, коли неможливо самому підписати якийсь документ, для онлайн-шопінгу тощо, потрібно адаптувати їх для сучасних постквантових криптосистем.

1) Розглянуто схеми цифрового підпису зі спеціальними властивостями у багатовимірних криптосистемах з відкритим ключем. Досліджено особливості побудови схем цифрового підпису зі спеціальними властивостями.

2) Побудувано загальну порогову проксі схему цифрового підпису з використанням примітивів багатовимірної криптографії. Ця схема може бути використана для побудови інших схем порогових проксі підписів з використанням примітивів багатовимірної криптографії.

3) Побудувано порогову проксі схему цифрового підпису на основі схеми підпису LUOV. Така значно зменшує кількість ресурсів, які використовуються для створення підпису групою користувачів, за рахунок спеціально підібраних параметрів, та значно пришвидшує створення цифрового підпису за рахунок реалізації арифметики у правильно підібраних скінченних полях. До того ж використання ідентифікаторів у схемі досволяє завжди мати уявлення про те хто приймав участь у створенні підпису.

4) Проаналізовано коректність і захищенність побудованих порогових проксі схем цифрового підпису. Оскільки для побудови цих схем використовувалося делегування з гарантією, ця модифікація ніяк не впливає на стійкість схем, які будуються таким чином.

ПЕРЕЛІК ПОСИЛАНЬ

1. Bernstein D.J., Buchman J., Dahmen E. Post-quantum Cryptography // Springer. [Електронний ресурс]. — 2009. — Режим доступу: https://www.researchgate.net/profile/Nicolas-Sendrier-2/publication/226115302-Code-Based-Cryptography/links/540d62d50cf2df04e7549388/Code-Based-Cryptography.pdf

2. Report on post-quantum Cryptography // NISTIR draft 8105, National Institute of Standards and Technology. [Електронний ресурс]. — 2018. — Режим доступу: http://csrc.nist.gov/publications/drafts/nistir-8105/nistir8105draft.pdf

3. Bogdanov A.,Eisenbarth T., Rupp A., Wolf C. Time-area optimized public-key engines: MQ-cryptosystems as replacement for elliptic curves? // LNCS 5154, Springer. [Електронний ресурс]. — 2008. — Режим доступу: https://link.springer.com/content/pdf/10.1007/978-3-540-85053-3$_4$.*pdf*

4. A.I.T. Chen, M.-S. Chen, T.-R. Chen, C.-M. Cheng, J. Ding, E. L.-H. Kuo, F.Y.-S. Lee, B.-Y. Yang. SSE implementation of multivariate PKCs on modern x86cpus // LNCS 5747, Springer. [Електронний ресурс]. — 2009. — Режим доступу: https://troll.iis.sinica.edu.tw/by-publ/recent/57470033.pdf

5. Ding J., Dubois V., Yang B.-Y., Chen C.-H., Cheng C.-M. Could SFLASH be Repaired // LNCS 5126, Springer. [Електронний ресурс]. — 2008. — Режим доступу: https://link.springer.com/chapter/10.1007/978-3-540-70583-3$_5$6

6. Dubois V., Fouque P.-A., Shamir A., Stern J. Practical Cryptanalysis of SFLASH // LNCS 4622, Springer. [Електронний ресурс]. — 2007. — Режим доступу: https://eprint.iacr.org/2007/141.pdf

7. Patarin J. The oil and vinegar signature scheme // Dagstuhl Workshop on Cryptography. [Електронний ресурс]. — 1997. — Режим доступу: https://core.ac.uk/download/pdf/455532201.pdf

8. Kipnis A., Shamir A. Cryptanalysis of the oil

and vinegar signature scheme // LNCS 1462, Advances in cryptology. [Електронний ресурс]. — 1998. — Режим доступу: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.120.9564rep=rep1 type=pdf

9. Kipnis A., Patarin J., Goubin L. Unbalanced Oil and Vinegar signature schemes // LNCS 1592, Springer-Verlag. [Електронний ресурс]. — 1999. — Режим доступу: https://core.ac.uk/download/pdf/205595201.pdf

10. Ding J., Yang B.-Y., Chen C.-H., Chen M.-S., Cheng C.-M. New Differential-Algebraic Attacks and Reparametrization of Rainbow // LNCS 5037, ACNS. [Електронний ресурс]. — 2008. — Режим доступу: https://eprint.iacr.org/2008/108.pdf

11. Yang B.-Y., Chen J.-M., Chen Y.-H. TTS: high-speed signatures on a low-cost smart card // LNCS 3156, Cryptographic hardware and embedded systems. [Електронний ресурс]. — 2004. — Режим доступу: https://iacr.org/archive/ches2004/31560371/31560371.pdf

12. J. Ding, D. S. Schmidt Rainbow, a new multivariate polynomial signature scheme // LNCS 3531, Springer. [Електронний ресурс]. — 2005. — Режим доступу: http://www.researchgate.net/profile/Jintai-Ding/publication/220271942-Rainbow-a-New-Multivariable-Polynomial-Signature-Scheme/links/5489ca100cf225bf669c73e4.pdf

13. Bleumer G. Blind Signature // Encyclopedia of Cryptography and Security, Springer.— 2011.

14. J. Ding, D. S. Schmidt Rainbow, a new multivariate polynomial signature scheme // LNCS 3531, Springer. [Електронний ресурс]. — 2005. — Режим доступу: https://eprint.iacr.org/2008/322.pdf

15. A. Petzoldt, A. Szepieniec, M. S. E. Mohamed A Practical Multivariate Blind Signature Scheme // Springer. [Електронний ресурс]. — 2017. — Режим доступу: https://eprint.iacr.org/2017/131.pdf

16. A. Petzoldt, S. Bulygin, J. Buchmann Selecting Parameters for the Rainbow Signature Scheme // LNCS 6061, Springer. [Електронний ресурс]. — 2010. — Режим доступу: https://www.researchgate.net/profile/Stanislav-

Bulygin/publication/220335916Selecting-Parameters-for-the-Rainbow-Signature-Scheme-Extended-Version.pdf

17. A. Hulsing, J. Rijneveld, S. Samardjiska, P. Schwabe From 5-pass MQ-based identification to MQ-based signatures // Cryptology ePrint Archive: Report 2016/708. [Електронний ресурс]. — 2016. — Режим доступу: https://huelsing.net/wordpress/wp-content/uploads/2016/07/20160630-darmstadt.pdf

18. Mambo M., Usuda K., Okamoto E. Proxy signatures for delegating signing operations // CCS. [Електронний ресурс]. — 1996. — Режим доступу: https://dl.acm.org/doi/10.1145/238168.238185

19. Jakobsson M., Sako K., Impagliazzo R. Designated verifier proofs and their applications // Advances on Cryptology. [Електронний ресурс]. — 1996. — Режим доступу: https://iacr.org/cryptodb/data/paper.php?pubkey=2526

20. Saeednia S., Kreme S., Markowitch O. An efficient strong designated verifier signature scheme // Springer-Verlag. [Електронний ресурс]. — 2003. — Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=683D6052F6A5D1034A0531733A29C69F?doi=10.1.1.58.9357rep=rep1type=pdf

21. Shamir A. ID based cryptosystems and signature scheme // Springer-Verlag. — [Електронний ресурс]. — 1984. — Режим доступу: https://link.springer.com/content/pdf/10.1007% 2F3-540-39568-7$_5$.*pdf*

22. Mu Huang X., Susilo W. Y., Zhan F. Short designated verifier proofs and their applications // Springer-Verlag. [Електронний ресурс]. — 2005. — Режим доступу: https://ro.uow.edu.au/cgi/viewcontent.cgi?article=11297context=infopapers

23. Sunder M., Vandani V. Identity based strong designated verifier proxy signature schemes // Cryptology eprint Archive. [Електронний ресурс]. — 2006. — Режим доступу: http://eprint.iacr.org/2006/394.pdf.

24. Sunder M., Vandani V. Identity based strong bi-designated verifier proxy signature schemes // Cryptology eprint Archive. [Електронний ресурс]. — 2008. — Режим доступу: http://eprint.iacr.org/2008/024.pdf.

25. Amit K-A., Sunder L. Proxy Blind Signature Scheme // Transaction on Cryptology Volume 2-Issue 1. [Електронний ресурс]. — 2008. — Режим доступу: https://eprint.iacr.org/2003/072.pdf

26. Bleumer G. Threshold Signature // Encyclopedia of Cryptography and Security, Springer. — 2005.

27. Boyd C. Digital multisignatures // 1st IMA Symposium on Cryptography and Coding. [Електронний ресурс]. — 1986. — Режим доступу: https://link.springer.com/chapter/10.1007/3-540-45961-$8_4$0

28. Silvio M., Ohta K., Reyzin L. Accountable subgroup multisignatures // 8th ACM Conference on Computer and Communication Security (CCS-8). [Електронний ресурс]. — 2001. — Режим доступу: https://www.cs.bu.edu/ reyzin/papers/multisig.pdf

29. Camenisch J., Michels M. Separability and efficiency of generic group signatures // Advances in Cryptology—CRYPTO'99, LNCS 1666, Springer-Verlag. [Електронний ресурс]. — 1999. — Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.5251rep=rep1 type=pdf

30. Gennaro R., Jarecki S., Krawczyk H., Rabin T. Digital multisignatures // Robust threshold DSS signatures, LNCS 1070, Springer-Verlag. [Електронний ресурс]. — 1996. — Режим доступу: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.32.1659rep=rep1 type=pdf

31. Rabin T. A simplified approach to threshold and proactive RSA // Advances in Cryptology—CRYPTO'98, LNSC 1462, Springer-Verlag. [Електронний ресурс]. — 1998. — Режим доступу: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.8348rep=rep1 type=pdf

32. Tolyupa E. An Algorithm of (n, t)-Threshold Proxy Signature with an Arbitrator // Моделирование и анализ информационных систем т. 20, №4. [Електронний ресурс]. — 2013. — Режим доступу: https://www.mais-journal.ru/jour/article/download/184/194

33. Chen J., Ling J., Ning J., Panaousis E., Loukas G., Liang K., Chen J. Post quantum proxy signature scheme based on the multivariate public key cryptographic signature // International Journal of Distributed Sensor Network. [Електронний ресурс]. — 2020. — Режим доступу: https://journals.sagepub.com/doi/pdf/10.1177/1550147720914775

34. Li H., Gao J., Wang L., Pang L. MPKC-based Threshold Proxy Signcryption Scheme // The International Arab Journal of Information Technology, Vol. 17, No. 2 [Електронний ресурс]. — 2020. — Режим доступу: https://iajit.org/PDF/Vol% 2017,% 20No.% 202/16961.pdf

35. Beullens W., Preneel B., Szepieniec A., Vercauteren F. Luov signature scheme proposal to the NIST PQC project (round 2 version) // Cryptology ePrint Archive. [Електронний ресурс]. — 2018. — Режим доступу: https://www.esat.kuleuven.be/cosic/publications/article-2874.pdf